M.Sc. Dissertation (COMM002)

Interactive Password Visualisation

Department of Computing

Gary Read

6183708

Academic Year 2015 - 2016

Abstract

This report documents the creation and use of an interactive visualisation of passwords to allow for the discovery and exploration of patterns in passwords. Here we learn from similar projects that use visualisation techniques, such as heat maps, to explore the role of dates and other semantic properties in passwords. The tool created in this report aims to visualise known patterns in passwords as well new, unknown patterns. This tool provides a flexible platform where patterns in passwords are discovered though good HCI concepts that aims to provide researches with a stand-alone, lightweight, functional, and modular tool that has been created with customisation in-mind. Building on knowledge of the semantic and other patterns in passwords that have been shown to improve the speed that password hashes are cracked, which could be used for fun, to demonstrate further weaknesses in incorrectly used and stored passwords, and be used in further applications such as profiling users and password policy development. This tool describes the development and use of the passwords visualisation tool which combines heat maps and parallel coordinates along side each-other. A new concept of relative password strength based on known passwords leaked is also born and implemented with the aim to provide users with a more accurate measure of password strength in the future.

Declaration of Originality

I confirm that the submitted work is my own work. No element has been previously submitted for assessment, or where it has, it has been correctly referenced. I have also clearly identified and fully acknowledged all material that is entitled to be attributed to others (whether published or unpublished) using the referencing system set out in the programme handbook. I agree that the University may submit my work to means of checking this, such as the plagiarism detection service Turnitin® UK. I confirm that I understand that assessed work that has been shown to have been plagiarised will be penalised.

Signed: 

Date: 29/08/2016

Acknowledgements

I must award this section towards my supervisor, Dr Shujun Lee, who has devoted much of his precious time into this project. Together we shared ideas and I learned much about research and about learning through his strict and direct approach to teaching. Without his enthusiasm for this project the results produced would have been of less significance, and I look forward to further developing ideas for this project with Shujun.

Contents

List of Figures

List of Tables

1 Introduction

Passwords are the still the main form of authentication for many services, however it is now well known that they do not provide adequate security alone due to multiple factors. Firstly humans, in general, has a bad habit of creating weak passwords. A study in 2007 found that web users on average have 25 online accounts and protect those with 6.5 passwords[1], giving evidence toward heavy password reuse. It would also be a fair assumption to say those 6.5 passwords some are mangles of some kind. Mangling is a technique used by users to create a memorable password from a word they already know, a name, a dictionary word, or password already in use. It involves substituting or adding nonces to the original string, such "123password" could become "123p4ssw0rd" or "password123". At first this might seem to be a good method of password creation and avoids password reuse and helps create memorable passwords for policy's, however password mangling is a proven technique used to crack passwords. Free open source tools such as John the Ripper and HashCat[1] implement mangling rules and there are even more complex rules being developed based on real passwords that have been leaked. There have however been attempts to strengthen passwords. Firstly by storing them correctly, the reader may have noticed the trickle of news concerning security breaches and dumped databases. Secondly, responsible parties should be educating users. Graphical password checkers and meters can be good to let users know how well there password fairs, making these checkers and meters reliable and based on actual evidence is the next task for the research community . Lastly, services should implement best practice policies, it is now becoming apparat that complex passwords policies are actually weakening passwords use. Bruce Schneier, an expert in information security, has mentioned this himself on his blog many times[2].

It is said a picture paints more than a thousand words, and the primary premise of visualising data. Data visualisation is done is many fields where there is a large enough set of data and where there are multiple dimensions or characteristics, and allows for organic and human centric method of pattern finding to further insights, teasing out meaning from the raw data.

> Data visualisation promotes creative exploration. - Simon Samuel, Head of Customer Value Modelling for a large bank in the UK

Visualisation techniques are not only limited to data comprehension and pattern finding, but it also helps communicate ideas effectively to us humans. Humans are not good at quickly comprehending information from raw data in tables, or even statistical analysis which aims to give meaning to data as not everyone is not a trained data scientist. An emerging trend on the internet is "information is beautiful"[3] where a focus is placed on educating people from all backgrounds, with sometimes complex data. The authors at informationisbeautiful.net have two visualisations of relevance to this project: World's Biggest Data Breaches, and Top 500 passwords.

This project wishes to combine the topic of password security, cracking, and visualisation in a way that allows the visualisation of passwords such that users can comprehend the data, finding patterns and meaning, with the goal of finding useful information that will aid in the education of policy makers, passwords creators (users), and help discover password cracking rules.

[1] http://www.openwall.com/john/
[2] https://www.schneier.com/
[3] http://www.informationisbeautiful.net/

The rest of this section is concerned with the high level aims and objectives of the project, the approach, and the structure of the rest of the document is given in the outline.

1.1 Project Aims

The aim of this project is to produce an application that can visualise passwords, in an interactive way to allow for the organic exploration of data. The solution should be concerned with the interactions between the tools and and user, so that it could be used with minimal training. The reason for attempting to create such a tool is that there seems to be actually very little research done in this area, which on an intellectual level, seems that it could provide some useful insights into multiple fields.

The solution should combine different visualisation techniques and have a flexible interface which allows users to configure exactly he components they wish, and it should also enable more advanced users access to the source code to allow for modifications or plug-ins to tailor functions to the users need, such as set operations to filter data. The whole system should be designed with modularity in-mind, be lightweight, and portable. The system should be a stand alone application with as little dependencies as is possible and run on multiple devices and platforms to cater for the distribution to interested parties who may not have the time or expertise to set up an application that requires pre-configuration.

In concise terms, these are the high level aims for the project:

- Visualise known password patterns better.
- Visualise new password patterns.
- Modular source code design.
- User configurable design.
- Low dependency on external products.

1.2 Project Objectives

The project objects describes the steps taken in order to meet the project aims. Visualising passwords so that it aids in interactive discovery requires knowledge of visualisation techniques, a deeper understanding of passwords and experience developing usable interfaces. Along with the research done a set of requirements gathered from potential users should be done to understand what features are actually desired, and how they should be coupled together with the interface. Lastly the system should be tested, comparing the results to the requirements and aim of the project, making recommendations where necessary.

Review known password patterns. It is important to understand what patterns already exist in passwords as the tool should aim to visualise at least some of these, or at least identify what type of patterns exist so that when using evaluating the system later it should be easier to comprehend the information presented by the visualisations.

Examine current methods in password visualisation. To help ensure the project is successful it is essential that lessons are learnt from previous attempts to visualise data and in particular passwords. Learning how passwords have been previously presented will help with the conceptual development of ideas, and in turn aid with the implementation and testing of the system.

Review visualisation techniques. For the same reasons as presented above, it is essential to understand the visualisation techniques that the systems aims to implement so that they are used effectively. Not every visualisation technique is appropriate, so it is important their strengths, weaknesses, and use cases are identified.

Identify requirements. Requirements gathered from users will help develop an system that users will want to use. Along side this, gathering requirements from the review of materials will also point the project into success and further research may help further iterations of the system.

Identify and review multiple concepts based on requirements. Designs will be created with the requirements and project aims in mind. The implementation of the system will depend on the ideas developed here. A justification and description of the designs is also necessary.

Implement solutions based on conceptual designs. Based on conceptual designs, user requirements, and knowledge gained though literature reviews a functional tool will be developed. Where the development process is monitored closely by select users who wish to use such a tool and are interested in the project. The feedback gathered will direct the next iteration and suggestions tracked for further development after the project has ceased.

Test and evaluate the system. The quality of the tool and of the software is to be evaluated through the use of metrics, users, or otherwise. The tests and results gathered here will access the functional and non functional aspects of the system, as well as the usability requirements defined. This section of the report will document the disparity between the final system and its requirements outlined earlier in the project and will give a measure of the successfulness of the project.

1.3 Project Approach

The project begins with a review of related literature into the related themes of this project: password security, password cracking, and human behaviour (regarding passwords). The review of literature gives the project meaning and position within computer science, and helps with understanding what patterns already exist in passwords and what patterns could exist, and a discussion of similar tools and password visualisations is conduced to learn from them. An analysis of the problem and the requirements of the system are discovered here by interviewing and working with security researcher and project supervisor Shujun. Functional, non-functional and usability requirements are captured at this stage.

Following the requirements and analysis conceptual ideas are created and discussed along with some diagrams to describe features and functions that help meet the aims and requirements for the

project. This process spawns further research into visualisation techniques and look closer at heat maps and parallel coordinates, which are the two techniques implemented later.

The implementation of the tool is discussed, documenting the software as it is. The system is then tested and a review of the results gathered to help assess how well the project met its aims and requirements. Finally a review of the project as a whole which concludes this project.

1.4 Project Outline

Literature Review

A critical review of related subject matter is required on the security of passwords, human computer interaction, and study what we currently know about passwords so that a tool can be created around these core concepts. HCI is crucial for the user experience and the effectiveness of pattern discovery while using the tool. A review of the current state of passwords is required so that we have minimal targets in what we which to show and discover with the tool.

Relevant Visualisation Techniques

A further review of materials and concepts are needed for this project. This section focuses on data visualisation, specifically looking into heat maps and Parallel coordinates. Each visualisation techniques have their own uses, advantages, and disadvantages, and this is where we discover and discuss this.

Analysis and Requirements

In this section an analysis of the functionality and usability required for this tool is gathered in order to create a list of core requirements for the password visualisation tool to be effective.

Methodology

This section outlines the design of the tool, the functions in which we may desire to implement. The final tool is built from these ideas however due to infantile stage and complexity of the research in this field the final tool differs somewhat. It is here where ideas are discussed at a high level.

Implementation

A description of the tool and all of its key features are described in section. The interface along with the functional aspects of this tool are documented with accompanying screenshots, code snippets and detailed explanations.

Testing and Results

Here we test the tool and evaluate it against the requirements defined at the beginning of the project. An assessment of the non-functional requirements are also studied, such as stability, usability and the other core requirements described in section 3.

Conclusion

This section summarises and concludes this report with an overview of the report as a whole. It will discuss how well the tool met its objectives defined at the beginning of the report. future work, recommendations, and a personal reflection with respect to the report.

2 Literature review

A thorough investigation into related concepts and literature is an essential part of seeing the wider picture in which this project sits. It is the knowledge gained from the literature review that allows this project to succeed and produce a successful tool that fits within the word of computer science. First an understanding of passwords, their use, how they are cracked, and how we currently view their security is required so that we can tailor the requirements of the tool toward this. This section later reviews several pieces of related work which contribute to password visualisation, further our understanding of passwords and how humans make them, and password cracking and policy making.

2.1 Password Security

The security of passwords is a contentious one. One one hand passwords are seen as a weak link, improperly used and not stored correctly. There are many policies that are soft and hard enforced on users which try to increase the strength and durability of passwords however, as documented later, it is shown that some policies actually have an adverse effect on the strength of user passwords, controversially some security professionals such as Bruce Schneier have said that users should not be forced to change their passwords every n-weeks as it encourages insecure behaviour on the users part, choosing a equally weak password that is simply an iteration of their previous password which is just as crackable, and where tools such as John the Ripper have rules built in that circumvent this kind of substitution technique in passwords. The human behaviour is reviewed in a later subsection.

One key area of interest for this project is to help find a method in which password strength can be calculated in a more accurate way, using empirical evidence from actual data along with traditional techniques such as the entropy of a password. The Strength of a password can not alone be calculated by its entropy, as later discussed, due to the fact password cracking rules ignore entropy and simply attack a probable known sub-set of passwords that has been developed over the years and increased with each password data breach.

One of the contributions that may come from this project is an improved idea of password strength estimation, coupled with a graphical password strength indicator. Dropbox in 2012[2] conducted research into password strength meters used around the web, as they developed there own. Password strength meters are used to help users choose stronger passwords and educate their users on their current password strength, which is best estimated by calculating the entropy of a password: length of password multiplied by log(c), where c = symbol space. However now that so many real leaked passwords are in the wild researchers have found ways to even high entropy passwords created by humans, due to the predictive nature in which we choose password. That is to say humans do not choose uniformly random passwords. Some of the most common passwords are “password1” and “7777777”. An analysis of the top 10,000 passwords showed that the top 10,000 passwords represent 99.8% of passwords used[3]. Due to the aforementioned analysis of leaked passwords, and others like it, password meters should really take this into account. Dropbox finds that strength meter implementations varied widely, and offered an unrealistic measure of the password strength. For example the password “correcthorsebatterystaple” is realistically a strong and usable password (easy to remember) however some services required a number, or a symbol, and others rated this password as weak. The conclusions were that password strength meters should take into consideration the spacial patterns, such as

"qwerty", they should take into account the entropy of a password, and they should take into account the known password frequency and dictionary words used in passwords, as well as any common password substitutions. Work done by Anthony et al [4] in 2013 look at the security of passwords when users are presented with some motivational/fearful information with respect to their password, such as a password strength meter saying "weak password" for example. The work found that interactive password checkers were much more effective than static password strength meters

Research into leaked passwords done by Matteo et al[5] try to evaluate the probability of guessing passwords in a password cracking session using the state-of-the-art approach. They find that the presence of password policies generally improved the strength of passwords, although as the attacks progressed they found these passwords created through policies were easily guessed, still. Their research concludes that even their is no one solution to cracking passwords and that it really depends on the user demographic and the password policy implemented. Further work trying to understand the make-up of passwords focuses on the semantics of passwords and the impact on security this has[6], this work was the first of its kind and presented a new method of cracking passwords even better than the state-of-the-art approach at the time. This work shows that there are still better way to compromise passwords by further studying the make-up of passwords and how users create them.

This project hopes that the information found by using the tool will help inform password policy makers. A password policies are used to impose constraints on the creation of passwords with the aim of increasing their strength. Examples of passwords policies might be; change every 3 months or it must contain symbols and capitals. The user of password policies have, under recent years, been criticised for actually creating weaker passwords as users end up choosing passwords which are there previous password with one character change, or create passwords which are easy to remember (and guess!) such as dictionary words[7].

2.2 Cracking Passwords

One key area of research still being conducted is the cracking of passwords hashes. Resolving hashes to plain-text passwords from leaked password databases gathered from the seemly never ending information breaches that corporations face, and with more and more powerful password cracking machines it is becoming easier to crack passwords, even high entropy ones in the near future[8]. Over the last few years, as more and more of us come online and create accounts for an increasing number of inter-connected services, we have increased the risk of being hacked. Cracking passwords is important in security as it shows weaknesses in protocols being used, and since the abundance of large password databases now available to researches it is possible to create rules to guess passwords more effectively, you could even say we can not guess passwords intelligently based on semantics and lexicons. The review of the following papers give a starting point of what this project should aim to help with, targeting the visualisation of known patterns and learn of other patterns that may remain to be discovered.

Most password cracking techniques are heuristic based, following a set of rules, a kind of educated brute-force guessing approach. As we learn more about passwords these heuristics become better at guessing passwords and password policies have been changes in order to prevent heuristic guesses by becoming more complex in retaliation. Research into the strength of these heuristics used compared with the use of different policies reveal that the effectiveness of attacks depends heavily on the policy

used, and the dictionary used (if a dictionary attack). A analysis of 70 million passwords from the Yahoo! data breach[9] found that password distributions did not vary much at all, with isolated patterns emerging which helps password crackers. Leaked passwords are helping improve password crackers. So much so that password cracking is becoming a script-kiddy activity, where any user can simply download a tool and have a high chance of cracking passwords, as documented by Nate Anderson in 2013[10] where it is shown how easy it is to crack passwords with the now available tools and knowledge of passwords. Further research is being done to further crack passwords by studying known passwords and using probabilistic techniques to improve guess rate of passwords[11], a wider study of 32 million leaked passwords further reveal common patterns in passwords and are shown to further improve password cracking techniques[12].

2.3 Human Behaviour

This section reviews literature focused on the choices humans make when creating passwords. The research here looks into why humans create passwords in certain ways how can we further use this information for predict passwords created by other humans. This project will be steered by the information learned here to help visualise these now known patterns.

Studies conducted on the user perceptions of security and their passwords have shown that most users knew how to make their passwords stronger, and which characteristics would increase the security of their password, which would make them harder to guess. This finding is surprising considering the pervasiveness of weak passwords used in the wild[13]. The work suggests that password meters should not only tell a user how strong a password is but also why it is weak, or strong. Another report[14] studied the creation of passwords in controlled conditions, step by step. They identified several algorithms users used to create passwords, some being quite secure, however most being insecure, showing that users have a varied view of security which identifies the need to educate users more on what security means for passwords and how they can improve it. In an attempt to improve the security of passwords strength meters are used, as previously mentioned. A study carried out comparing strength meters[15] showed that they were useful and increased the security of the passwords used and helped create password that were not only more secure but were also just as memorable. It was also shown that more rigorous strength meters did create more secure passwords.

A study into the attitudes and behaviours of users during password creation, comparing differing policies, showed that users were annoyed by strict policies, and that some users struggled to create password at all[16]. The study showed that are more likely to share a password the longer they have used it, and that users would use old passwords modified to meet requirements. These findings match those earlier identified, and also backed up by a study of passwords gathered from users from a university. The results[17] learnt from users at a university showed differences in password security depending on what a user studied at university. Users who studied computer, IT, or maths related fields would in general pick stronger passwords where users which studied subjects such as biology, and physiology were more likely to choose passwords which were not secure. This research further adds to the value in researching patterns in passwords so to be able to identify demographics and other such factors in passwords to help cracking and education of passwords and their security.

2.4 Password Visualisation

The last literature review of this section focuses on the work specifically related to the visualisation of passwords and the identification and analysis of patterns in lists of leaked passwords. The goal here is to learn from these successful projects and aim to produce a tool which can at least replicate some of these features or functionality in a way that can be done in a interactive sense.

2.4.1 PIN analysis

A blog entry by Nick Berry at DataGenetics in 2012[18] explores the distribution of PIN numbers, taken from various leaked sources online which are not disclosed. The total data set consists of 3.4 million 4-digit PINs. The goal of this research was to find, if any, patterns in 4 digit pins to educate the security community as well as inform security conscious users.

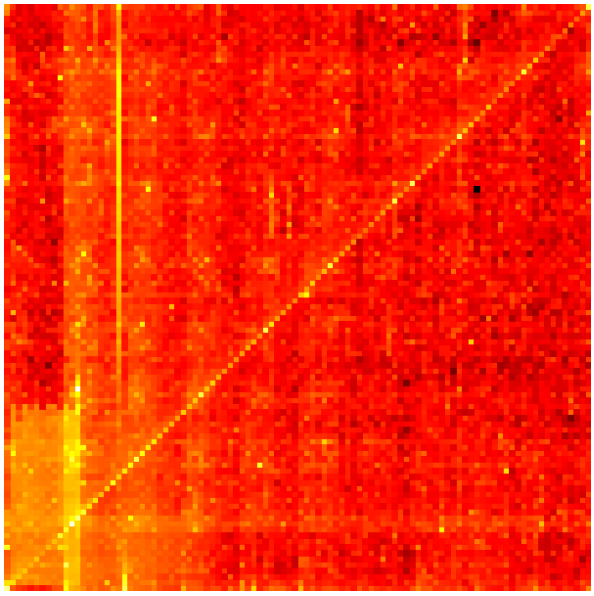


Figure 1: Heat map visualisation of PINs by DataGenetics

The research conducted focuses on 2 aspects, visualisation and statistical analysis of the data. The statistical analysis reveals some shocking facts about user chosen PIN numbers, which are not uniformly random! Nick shows that of the possible 10,000 possible combinations of 4 digits (between 0000 and 9999) a shocking one-third of PINs will be guessed by trying only 61 PIN combinations, and that the top 50% of PINs are represented by only 426 distinct PIN numbers, not 5,000 which would be expected for uniformly random PINs. The report also identifies the least common PINs however Nick correctly notes that these PINs might become more common now that this study has been released, 8068 is identified as the least common PIN with a frequency of 25/3,4000,000. Statistical analysis also identifies that a large percentage of PINs begin with 19??, indicating that many users use 20th century dates as there PIN, perhaps their birth year or that of a family member. Plotting a histogram of all the years between 1900 and 1999 it is believed to give a good measure of user demographics, which is well justified.

Further patterns are discovered in the data by way of heat map visualisation. Plotting the left two digits against the right two digits as the x and y axis respectively from 00 to 99, and represents every PIN combination. This visualisation can be seen in figure 1. The brighter (white/yellow) a cell is the more frequent that combination appears within the data, and at first glance the reader should notice 3 key features: the diagonal line, the bright linear line top left, and the large block bottom left. The reader should also notice the odd outliers scattered around.

Nick finds a few interesting things, which are summarised below. An almost hidden feature is the grid-like feature within the plot, this is because users tend to choose numbers that are close to each other such as 45 and 67 rather than 73 and 37. There are some exceptions to this rule of course, and one of them relates to the bright line top left down the y-axis. This line represents years in the form

19?? which was discussed earlier. The bright outliers spattered around were identified as numerical runs such as 2345, 4321, and 5678. The other obvious feature is the diagonal line representing PINs in the form XYXY which represent almost 18% of all the 4-digit PINs. One last observation worth mentioning is the lower left of the heat map. It was identified by a reader of the blog (zero79) that this section is the encapsulation of dates in the form MMDD (i.e. month then day).

Lessons learned from this blog entry by Nick has provided some patterns which this project should try to visualise. It has also given a general sense towards to semi-predictive nature of humans and their password choices.

2.4.2 Visualising Passwords using heat maps

A project conducted by student Yin To Lee[19] in 2015 attempts to create a password visualisation tool that is configurable, easy to use, and allows modification of the heat map. The tool can be seen in figure 2. The password visualisation is done with the help of the d3.js library, a data driven visualisation library, and is written entirely in JavaScript and HTML so that the program is compatible for a variety of devices and has little dependencies.

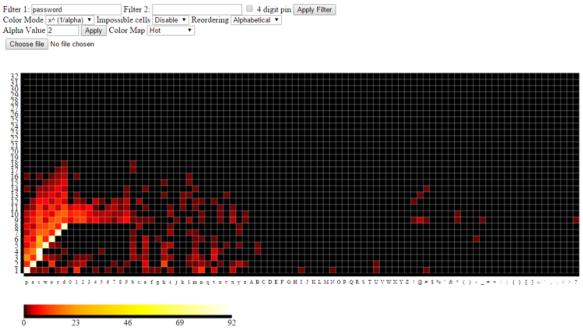


Figure 2: Password visualisation tool by Yin Lee

The tool has a few functions to provide a better analysis of password patterns, including filters, colour distribution function, heat map colour scheme, reordering, and 4-Digit PIN mode. Referring back to figure 2, the heat map shows passwords containing the word “password”, and is ordered by the search word and then alphabetically. This visualisation shows the distribution and frequency of characters that follow on from “pass” and “password”, as well as showing the different mangles of “password” shown by the high frequency above the brightest line representing “password” starting at [p,1]. The colour mode allows for the amplifying of low or high frequency data, which can help tease out patterns. Another feature of this tool is the legend and the cell tooltip containing information about that cell, frequency, position, and the character.

This project makes an interesting step towards an interactive password visualisation tool which this project will learn lessons from.

2.4.3 Visualization and Estimation of Empirical Passwords

Research done by Guo Xiujia et al. titled “The scale-free network of passwords: visualisation and estimation of empirical passwords”[20] develop techniques to visualise empirical passwords as networks with the aim to reveal connections between passwords, such as “123456” and “a123456” are strongly related, as shown by the size of the passwords in figure 3. Guo et al. also then propose a statistical model to aid password cracking with “educated” guesses based on statical data produced directly by their research.

They are the first to discuss and produce spacial representations of password sets, the visualisations produced show how connected a password is in respect to other passwords already existing in a set, this

work could be used to create graphical password checkers that encourage users to choose passwords that are disconnected from highly connected passwords. The statistical side of their research develops a guessing model which gives the number of guesses required to crack a percentage of passwords.

2.4.4 Visualising Semantics in passwords: The role of Dates

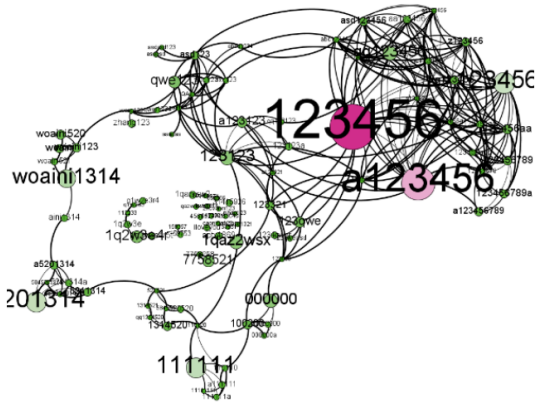


Figure 3: Graph of empirical passwords by Guo Xiujia et al.

Perhaps the most work done in the field of password patterns and semantics in done by Radael Veras et al. in their paper titled "Visualising Semantics in Passwords: The Role of Dates"[21] and in their related titles. Their report focuses on the role of dates in passwords, and come up with an interactive visualisation to explore these. The paper continues from their research into the categorisation and identification of password semantics. They find that users like to use words such as bad[??] and good[??] as well as pornographic related passwords. In this paper however there is a focus on dates. The authors also comment on password policies stating that 5% of the password set (RockYou) studied are dates, which lends a good reason to perhaps encourage users to not use dates in their passwords as date patterns are easily created to help crack lists of passwords.

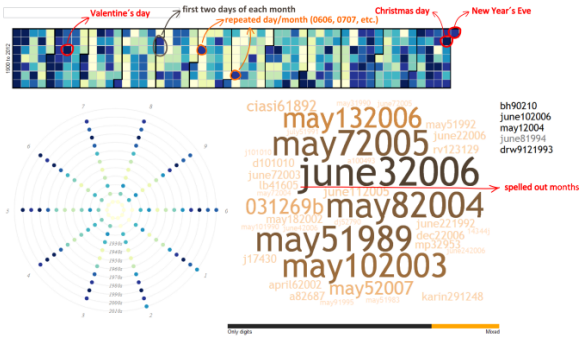


Figure 4: Visualising semantics in passwords: The role of dates by Rafael Veras et al.

They turn to visualisation due to the fact that automatic discovery of password semantics is a difficult task and that by means of visualisation more patterns may present them selves, especially when interactive. Their observations from visualisations revealed which date formats were preferred, as well as which time of year, and months were more popular. Holidays are discovered to be popular as the reader might expect, as well as previously identified patterns in the PIN number report at DataGenetics[18] where repeating month and day were popular (i.e., 0707 and 0808).

The visualisation they come up with is especially unique as they choose 4 different but interconnected methods of visualisation. Heat map to visualise the distribution of MMdd formatted passwords. A radial heat map plot to visualise the distribution of years in passwords, a Wordle scaled on frequency

of the password chosen, and finally a bar char to visualise the distribution of digit dates and mixed formatted dates. The research done here gives this project some good ideas about user interaction and what kind of visualisations are useful for pattern discovery.

2.5 Summary

This section has covered many highly related topics from which this project will learn from. The placement of this project is well placed within the security of passwords, as well as password visualisation. The review conducted here also helped scope the breadth of this project as well as generating future ideas relating to further research to do with the visualisation of passwords to help with the generation of password policies, password strength meters for education, and to help researchers crack passwords better.

3 Relevant Visualisation Techniques

Along with the review of literature relating to the core concepts of this project, it is necessary to look further into the actual visualisation of data. An in depth discussion of two visualisation techniques: heat maps and Parallel coordinates. This section will review their advantages and disadvantages, when they to be used, as well as why they work.

3.1 Poor Visualisation Practices

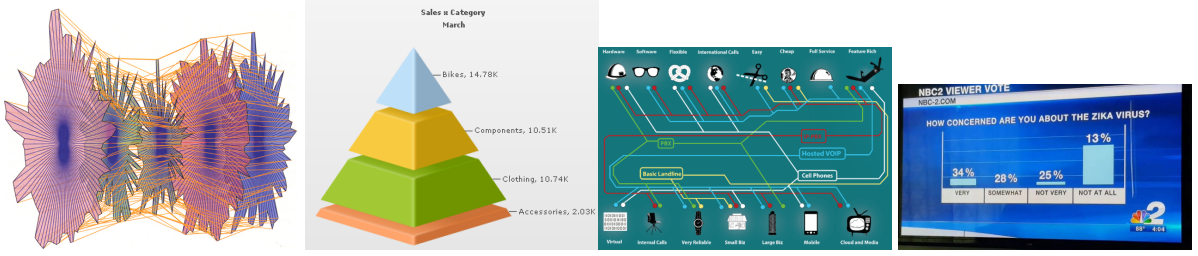
Visualisations are created to help comprehend data so to extract meaning from mere numbers, however there is a fine balance between a good visualisation and bad one, in which the later does nothing but confuse users and ignores the data for visuals. This project must present visualisations of password so that users can easily understand what data is shown and find some meaning from this, to this end it is appropriate to review examples of bad visualisations and learn from them.

- Figure 5a uses pseudo 3D with overlapping spike charts and parallel coordinates. This visualisation completely fails to show any data. Firstly there are no labels and no key. The pseudo 3D spike graphs mean that they overlap and cover other parts of the visualisation. It is impossible to view the parallel coordinates on the back side of the visualisation. Along with this the parallel coordinate are made useless by the fact they have been rotates around the spike graph.

- Figure 5b shows a 3D pyramid chart. The key thing wrong here is that the visualisation is confusing volume with height. Volume is difficult to compare, especially considering the non-uniform shapes being used. It is quite impossible to understand the meaning of this chart where the two middle slices are given the same value however the bottom is clearly more in volume.

- Figure 5c shows a confusing visualisation. The lines overlap and there is no order. Visualisations should help users understand the data quickly, not confuse them.

- Figure 5d is a bar graph shown on US national television by NBC. The visualisation has no concept of scale here where the it seems 13% is greater than 34% of users? Very confusing and misleading. The goal of visualisations is not to mislead but to quickly convey the meaning of data.

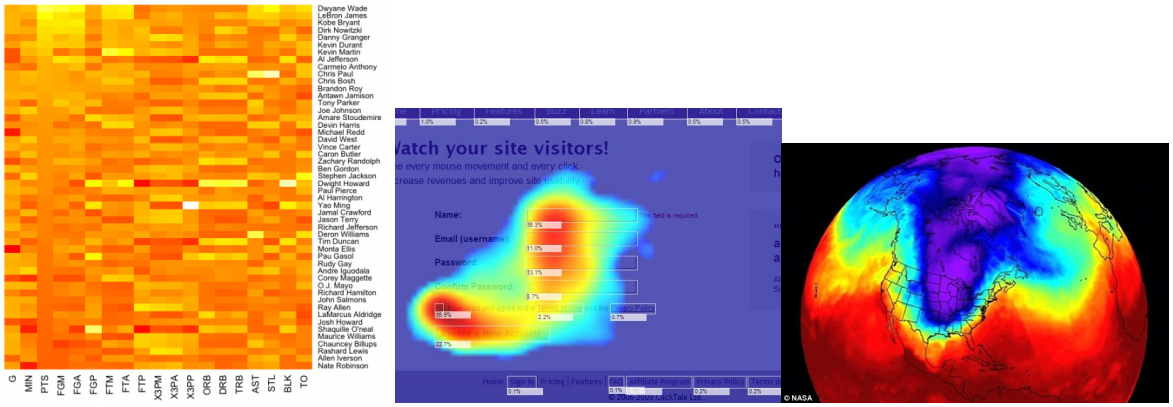


(a) Spike chart with Parallel Co-ordinates (b) 3D pyramid chart (c) Linked diagram (d) Bar chart presented by NBC news

Figure 5: Examples of poor virtualisations

3.2 Heat maps

One of the visualisations chosen for this project to visualise password is the Heat map. Heat maps are able to visualise 3 data dimensions clearly on a 2D plot, x and y axis represent two dimensions and the third dimension is the colour allocated to a xy coordinate, the colour known as the “heat”. Figure 6 shows three types and use cases of heat maps. The first is a classic, mosaic, plot. It is used to represent a two-or-more way table. The second example is called a web heat map, and uses continuous colouring method to visualise the time a user spends over parts of a web page. The last example is a composite visualisation, combining a geographical map with a continuous heat map visualising the level of Ozone over different locations around the world, this method is also common for weather visualisations. We choose to use heat maps as it has the ability to visualise the distribution of characters in passwords and their frequency[22].

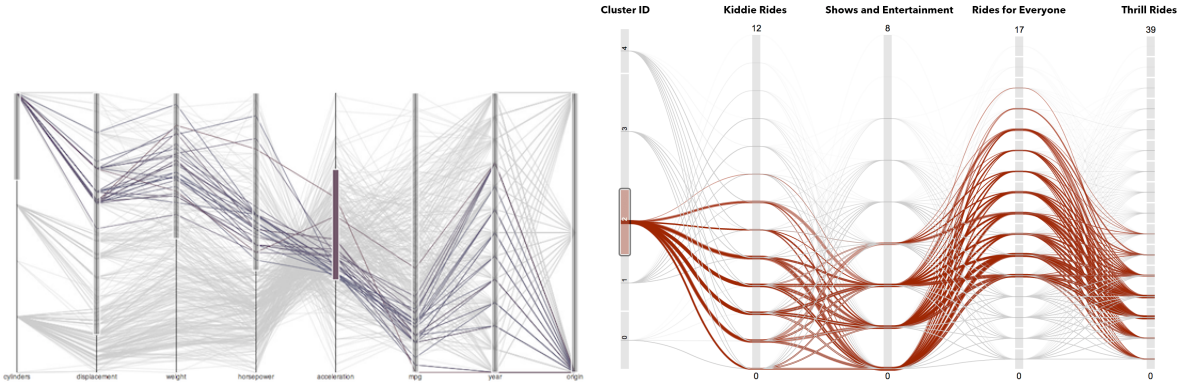


(a) Matrix heat map (b) Continuous heat map (c) Continuous heat map with geographical overlay

Figure 6: Different types of heat map

3.3 Parallel Coordinates

Parallel coordinates is a visualisation technique to plot individual data elements over multiple dimensions, infinitely many in-fact[23]. Each vertical axis represents a dimension. The x axis is used to represent the relation between dimensions via the “slope” of the line between points (See figure 3.3) and is a good visualisation technique to visualise high dimensional data. The limitation with these though is that they can become cluttered and difficult to read, attempts such as line clustering shown in figure 7b try to help keep the visualisation readable by clustering lines from the same point, or by separating lines over the labels[24]. This project wishes to use parallel coordinates to try and visualise any relationships between characters and their position, as well as visualise passwords against more complex dimensions such as their semantics.



(a) Parallel Coordinates (b) Parallel Coordinates with Clustering

Figure 7: Different types of Parallel Coordinates

3.4 Summary

In summary in this section a deeper understanding of visualisation techniques required for this project are gathered. The problems that must be solved will greatly benefit from an understanding of the core concepts of visualising data, from the user experience to finding the balance in displaying multiple dimensions and useful information still being inferred from the data. Later in this project a discussion on using heat maps with parallel coordinates are made, and what solutions are created, discussing their limitations and complications.

4 Problem Analysis and Requirements

This section is concerned with the gathering and analysis of requirements for the project. The tool being developed has a large focus on the usability and interactivity between the user and data, it is for this reason that it is essential the requirements for the project are discussed and taken from potential users of the tool. The functional and non-functional requirements are outlined along with the usability requirements for the tool to be successful. The decisions made here are detailed and given a justification.

4.1 Requirements Gathering

The requirements for this project are mainly based on two factors: What the user wants and what the is learnt from reviewing related literature. The review of literature was conducted in section 2, and so here the gathering of the user requirements are necessary. This project communicates its ideas directly to security researcher Shujun, also project supervisor, who is asked questions and gives feedback on ideas created here. Several interviews with Shujun reveal what kind of feature a password visualisation tool might want to have, combined with the knowledge gathered in the literature review a list of function, non-functional, and usability requirements can be made which combine both the information learnt from Shujun (product user) and the literature. The reader should note that due to the nature of this research the requirements for this project may have changed later in the project, as new features are developed, ideas may not be feasible, and the project could change direction depending on the results obtained from the initial tests of the first implementations.

4.2 Functional Requirements

Feature of the software are described by its functional requirements, what the system provides to the user and what it should do. Each requirement is given a priority so that the most important features are at least present due to the imposed time and resource constraints of this project. Table 1 provides an overview of the functional requirements and its priority.

No.	Description	Priority
1	The system allows for data to be manipulated.	Must have.
2	The system visualises data.	Must have
3	The system allows for customisation.	Must have
4	Visualisations are interactive and responsive.	Must have
5	The system supports multiple file upload.	Should have
6	The system supports multiple file formats.	Should have
7	The system notifies the user during long processes.	Should have
8	The system allows for the comparison of visualisations.	Should have
9	Statistical information is presented by the system.	Could have
10	Graphical password checker.	Could have

Table 1: Functional requirements

4.3 Non-Functional Requirements

The non-functional requirements of the system propose constraints on how the design and implementation of the system, describing how well functions are to perform. The requirements describe the systems usability, compatibility, and availability. Table 2 lists the non-functional requirements for this project.

No.	Description	Type
1	The system should be intuitive and self explanatory.	Usability
2	The system should be easy to maintain and be modular.	Maintainability
3	The system should be developed for the web.	Compatibility
4	The system should limit use of external libraries.	Portability
5	The system should be compatible with multiple devices.	Portability
6	The system should process queries within a responsible time.	Usability

Table 2: Non-Functional requirements

4.4 Usability Requirements

For the project to be successful in terms of user discovery of patterns it is important the tool follows the best usability principles. This project attempts to follow the tried and tested heuristics developed by Jakob Nielsen in 1994 [25]. These are broad rules that help developers create applications applications. The list of heuristics is shown in the table below.

No.	Heuristic	Description
1	Visibility of system status.	The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.
2	Match between system and the real world.	The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.
3	User control and freedom.	Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.
4	Consistency and standards.	Users should not have to wonder whether different words, situations, or actions mean the same thing

5	Error prevention.	Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.
6	Recognition rather than recall.	Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.
7	Flexibility and efficiency of use.	Accelerators – unseen by the novice user – may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.
8	Aesthetic and minimalist design.	Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.
9	Help users recognize, diagnose, and recover from errors.	Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.
10	Help and documentation.	Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Table 3: Nielsen's usability heuristics

4.5 Summary

This section discusses the process of the gathering and analysis of the requirements for the tool. The functional requirements for the tool are outlined here, as well as the non-functional requirements which set guides on the implementation and design of the tool. Finally a review of Nielson's usability heuristics is conducted. The information learned during the analysis phases greatly aids the success of the project as the requirements are specified from the beginning, from the users of the system and from related systems from the literature review.

5 Methodology and Design

This section focuses on the methodology used in the development of the tool as it is important to plan ahead in order to maximise the project meeting its aims and requirements. Following on from the development method a discussion of ideas is had to meet the aims of the project. These ideas are all conceptual and not all will be implemented due to multiple factors however it is important to note these down to learn from and for future development to continue. The final part of this section focuses on a high level overview of the system proposed and a discussion on the experiences developing is had.

5.1 Development method

The development method used here is a rapid prototyping method, where the requirements for the project could change with each implementation it would be a poor use of time to fully develop the system before implementation. As the research here is limited it is not really known what concepts will work or how they will be implemented given computational and resource constraint; such as memory and time.

5.2 Design pattern

To ensure code is maintainable it was chosen to use the modular revealing design pattern. This design pattern allows for the modulation of JavaScript code such that variables are scoped only to their class/-module, which helps prevent bugs and eases maintenance and documentation. The revealing feature of this design pattern is that to reveal a function for public use it must be explicitly added to the return object of the class/module, this allows for a developer to quickly identify what functions are public facing and which functions can be interacted with by other objects. The following code snippet is an example of the the modular revealing pattern.

```
var Calculator = function() {

    //Internal function
    var square = function(n) {
        return n * n;
    }

    //Revealing functions to outside
    return {
        getSquare : square
    }
}
```

5.3 Concepts

5.3.1 Heat map/Parallel coordinate interaction

Interlinked visualisations as used in Veras' paper; The role of Dates[21], appear to be a very powerful method of visualisation. Where interacting with one modifies the other. The idea here is the inter-

activity between a heat map visualisation and a parallel coordinate plot of the data set, as shown in figure 8. The two types interactions would be:

- User selects heat map region where the parallel coordinate plot updates and only shows those password within the region created in the heat map.
- User selects subset of a dimension, or multiple dimensions, and the heat map is updated to visualise only those passwords included in the selected subset.

This kind of interaction should allow for the exploration of complex patterns in the semantics of passwords, however coming up with a good method of determining dimensions for the parallel coordinate plot may prove to be difficult as this most requires the use natural language processing to tease out meaning from the passwords.

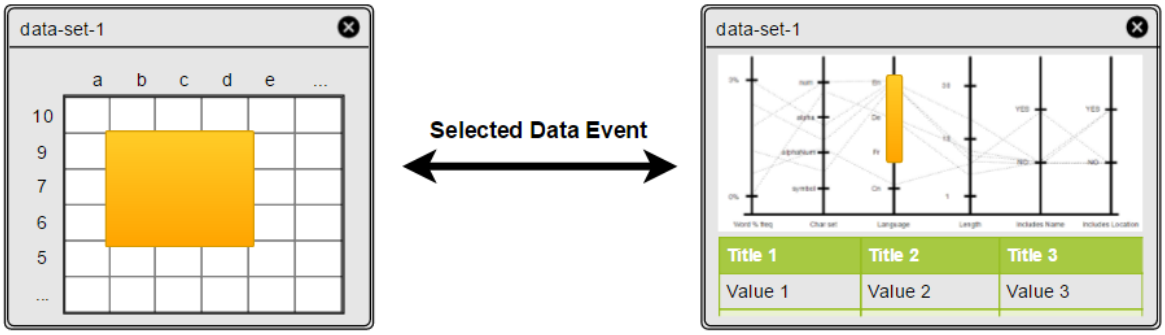


Figure 8: Parallel Coordinate and Heat map user interactions

5.3.2 Combined heat map parallel coordinate visualisation

Combining visualisations may help show patterns or give meaning to unknown patterns in passwords. Figure 9 shows the concept of overlaying parallel coordinate lines over a heat map visualisation of the password set. Each line represents a word, and each dimension of the plot is the position in the word (a parallel coordinate plot turned 90 degrees). Again it is unknown how well this will preform with real data. It is assumed that there may be too many lines, over 7 million for the RockYou list, and any information would be hidden. There is one idea proposed by Shujun, which is difficult to compute and so will not attempted in this project, which is to highlight sections of parallel coordinate lines that appear frequently in other parallel coordinate lines. For examples the lines representing the words “password”, “213password321”, and “1password!” the line segment covering “password” in each line would be brighter. However identifying these patterns in words is a big computational task and beyond the scope of this project.

5.3.3 Filtering

Inspiration from web shopping web sites such as Amazon and Ebuyer gave rise to this concept of filtering shown in figure 10. The idea here is categorical filtering for the semantics in passwords, and the stacking/combining of searches. Such that a user may be able to filter for cities in England AND languages, or a use may choose to filter simply for all geolocations. Now, this would certainly be a powerful feature however the author does not ignore the complexity of the problem at hand, which is

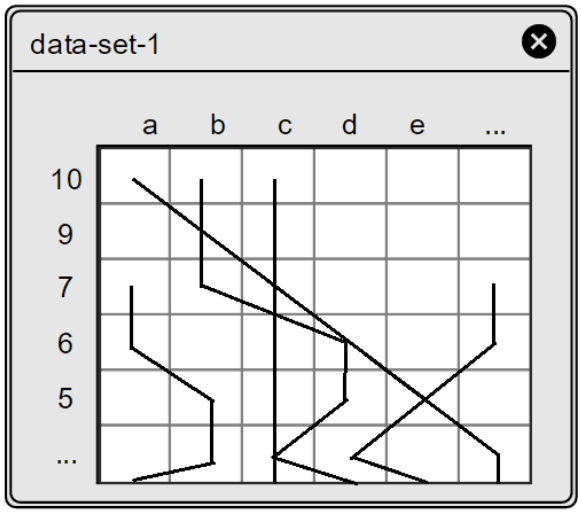


Figure 9: Combined visualisation of Parallel Coordinates and Heat map

the semantic categorisation of words within passwords, which are two separate and difficult problems which are solved by natural language processing which still a topical research field.

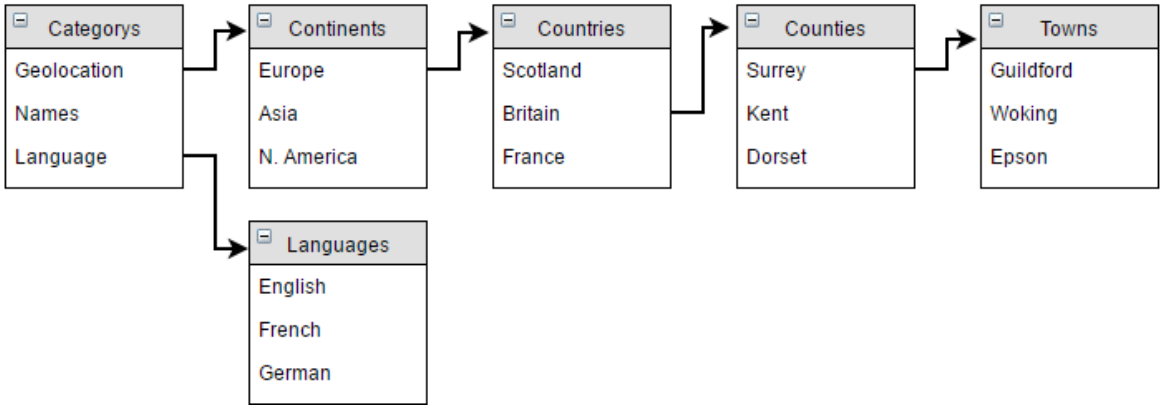


Figure 10: Drop down menu - Semantic filters

A more realistic solution for this project is to enable the manual filtering of passwords by lists of words or using set operations on password sets with a list of words. Easy to implement filters such as date, PIN, and words including should be implemented at the very least to prove the usability of the visualisation techniques. Once this has been shown further implementation can be made to filter for these semantics such as the research done by Veras [6].

5.3.4 Parallel coordinate visualisation of semantics

Continuing on from filtering by semantic categories of passwords it would be useful to be able to visualise these. Two concepts for visualising this have been developed.

1. Visualisation of general semantics and general properties to do with the whole set. See figure 11.

2. Visualisation of specific semantic property, such as dates. See figure 12.

The first concept allows for the user to identify key characteristics of the password list quickly, with the intention that they would compare and then further filter the data based on the findings from the parallel coordinate plot of general statistics and broad semantic categories such as language or location.

The second concept allows for a deeper look into the data, looking at the make up of specific semantic properties, such as dates. In the figure mentioned it is imagined that a user could choose the dimensions of the plot to help find relations in the data, such as those found in Veras' study of dates [21].

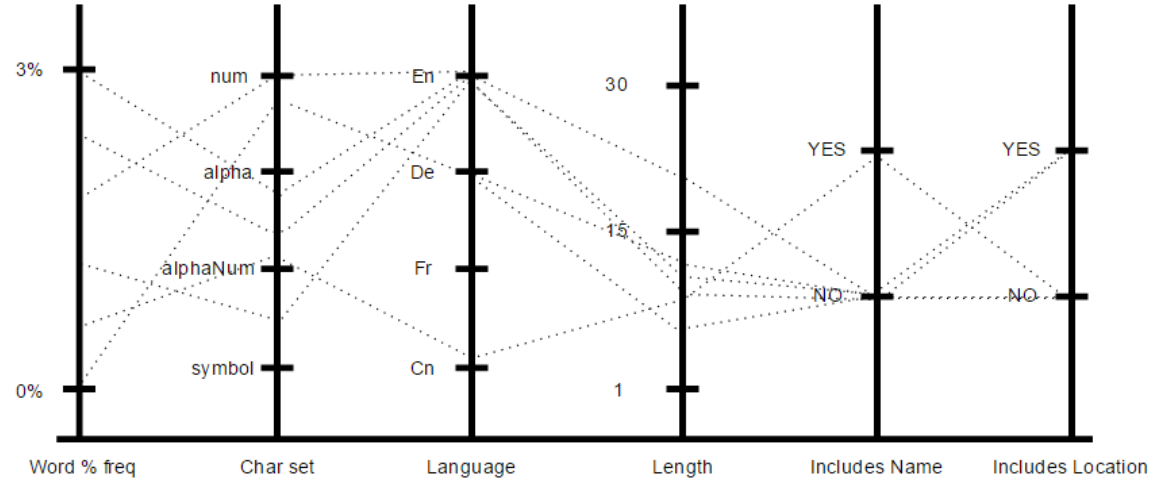


Figure 11: Slope graph visualisation of overall semantics of a data set

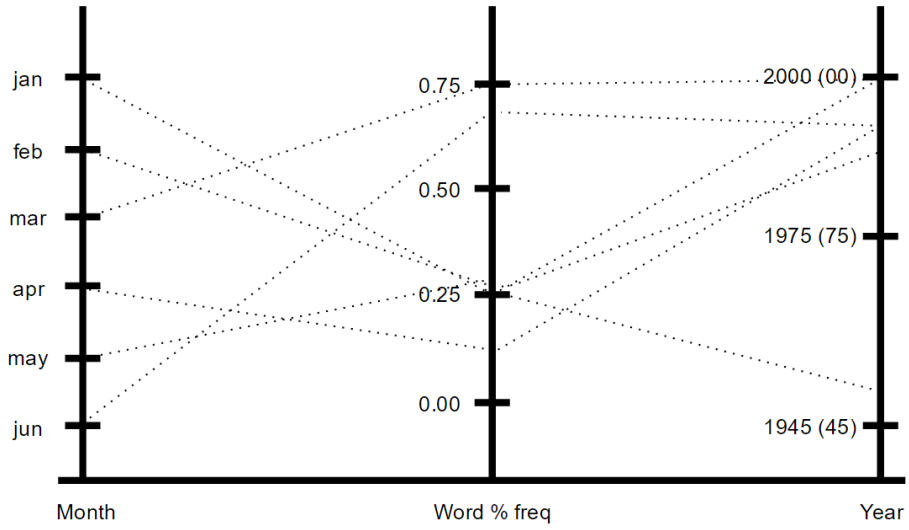


Figure 12: Slope graph of specific semantic category

5.4 High level system overview

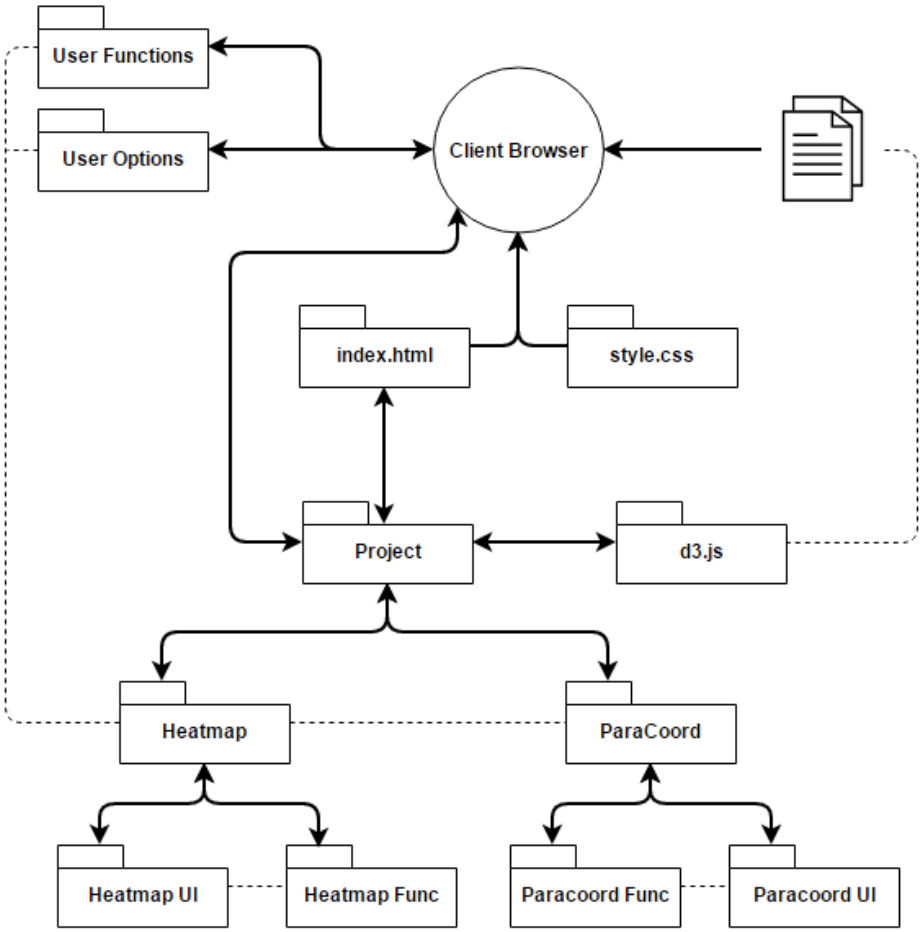


Figure 13: High level system overview

Due to the rapid prototyping method of development mistakes were made during the first two iterations of development. In the first stage of development the author attempted to build on from the work from Yin Lee, whose code did not have maintainability in-mind. An attempt in using a previously existing library for the visualisation of parallel coordinates was also a mistake as due to the nature of this project being quite unique a custom solution was going to be quicker. The second iteration of design started from scratch, both for parallel coordinate and heat map visualisations, with only d3.js as a dependency, which produced a well written and functional tool until new ideas were created during the use and development of the prototype. The requirement for a history function to be available required greater separation and modulation of the code, which required a whole re-write of the 2nd prototype, with this process the author opted for a proper review of the classes required for the software to meet its aims and requirements. The high level design approved can be seen in figure 13.

The final design of the system, the 3rd prototype, consists of a main project which contains references to multiple Heatmap and Paracoord classes, one for each file loaded in by the user. Each visualisation class has reference to a functional class and a user interface class, this separation allows for logical placement of functions and maximises the maintainability. The main class, Project, would contain any functions shared by the Heatmap and ParaCoord classes. The main Project class is the main calling point for the browser, where index.html called its main function on page load.

The user options, configuration, and function files are externalised to allow for the flexibility required by the tools aims.

5.5 Summary

This section discussed design concepts in detail, highlighting concerns with implementation and justifying the ideas against the aims of this project, presenting some evidence to back these up from the literature review. A high level design is created which is used in the implementation section 6 on the facing page.

6 Implementation

This section now describes the implemented and functional tool as it is. Firstly discussing design choices in its design and the choice of platform targeted with justifications as well as a discussion of the tools used during the development stages of the tool. The functions that have been implemented are listed and described here too.

6.1 Platform and Libraries

During the requirements gathering processes the study revealed that the tool should be targeted at as many devices as possible, suggesting a web based solution, with as little dependency on libraries and if possible remove the need for a virtual machine environment such as with Java and the Java Virtual Machine. Though there are advantages in using libraries and advanced programming languages such as Java they may cause limitations later on in development, namely due to the new nature of this research and the fact that there are high level requirements for this project. If this project were to create dependencies on any libraries these libraries may become a source of hindrance in the future if, for example, the research takes a different approach and the library is no longer relevant, or that the library is often updated which may cause bugs to appear and involve constant maintenance, and considering that one of the objectives for this project is that the source code must be maintainable it may not meet its requirements.

Google Chrome is a web browser released in 2008 by Google. It has implementations on Windows, Android, iOS, OS X, and Linux. Google Chrome is also based heavily on the open source Chromium project and which shares approximately 50% of the market share across all platforms and over 62% of desktop use (according to StatCounter[4]). This project targets development for use with The Google Chrome browser due to its open source nature, and the fact it is so widely used across al major platforms. Chrome rendering engine is built, Blink, is built from the WebKit engine which is important to note due to the fact that different browsers implement different engines and such the layout and usability of the tool is dependant on the engines interpreting the style sheets correctly.

This project attempts to reduce the dependencies on libraries, such that this project is only dependent on the d3.js library. d3.js is a library which contains a hoard of useful functions which aid in the development of visualising and processing data, it is popular and widely used library. It has a similar syntax to that of jQuery, is well documented, and has many examples created by the open source community available on the web, thus this library is used due to these attributes.

Development platform , languages, and libraries

A full list of languages, libraries, and platforms used in this project.

- JavaScript / ECMAScript - version 262 3rd edition (at least)
- HTML5
- CSS3 (Webkit support only)
- SVG

[4] http://gs.statcounter.com/#desktop-browser-ww-weekly-201501-201627

- d3.js - version 4.2.2
- date.js

Note: There is only a lose dependency for WebKit supporting browsers, the tool will work on other browsers, however the reader should expect some buggy or unpleasant behaviour while using the tool. A future fix will be to add compatibility for Microsoft Edge and FireFox.

Note: The use of date.js is a lose dependency and is not required for the program to function with exception to date filtering. It is intended that the date filtering ability to be expanded beyond date.js at a later date.

6.2 Development Tools

The tools used during the development of this project were not set in the requirements though it may be useful the the reader to know as this document is aimed at researchers and students who may wish to continue the project. In this sense it will be useful to know what tools were used and why.

For the development of JavaScript code a mixture between Notepad++ and Google Chrome's developer console (F12 in Google Chrome) were used throughout. The developer console within Chrome is a powerful tool with many features to analyse and test web applications, such as device compatibility, performance testing, debugging, and interactive code development. An especially good feature supported by Chrome's developer console is the ability to change element styles by selecting elements either by selecting DOM elements in he source or by selecting rendered elements. This allows for the quick debugging of complex layouts. A disadvantage with the developer console is that the source code editor is not particularly powerful, in a sense it does not support many common short-cuts for quick development of code. That is where Notepad++ comes in.

Notepad++ is an open source application which supports plug-ins developed by the open source community. Through experience this is the writers preferred free coding tool. Supporting VIM like features for line and word editing, source code can be quickly be edited and explored. Making use especially of the split window feature to view multiple classes simultaneously and with the support of plug-ins such as auto-documentation, bookmarking, and code completion for functions and variables.

Testing the Tool required the use of a local server due to security constraints in web browsers that do not allow direct access to the local file system, and since the application is required to visualise word lists given by the user it is necessary to obey these conditions. XAMPP[5] is a free to use Apache, DB, PHP and Perl development environment, it is a self contained web server that is easy to configure and set-up. To host the application see the instructions in Appendix A.

6.3 System Components

Guided by the requirements analysis, what the project wants to achieve and the design stages of this project a collection of system components can now be documented. Here each subsection covers a part of the system and documents its parameters, usage, and any other discussions that may be necessary. The full list of components below followed by a detailed review of each.

- Short hand UI functions.

[5] https://www.apachefriends.org/index.html

- Visual customisation.
- User configurable options.
- Filters.
- Drag & Drop.
- Normalised dataset merge.
- Cross Correlation Coefficient.
- Heat map history.
- Heat map statistics.
- Character histogram.
- File import.
- Synchronised UI.
- Pop-out configuration pane.
- Graphical password checker.
- Parallel coordinate overlay.
- Flexible & fixed display.

6.3.1 Short hand UI functions

In order to help allow advanced users and with the general feel of the application there are features implemented to increased the usability for advanced users while simultaneously for novice users, and without impacting the usability for either u

General sidebar tab The sidebar holds elements to manipulate the data of selected heat maps. To further help users customise and select heat maps a general section was created, seen in figure 14. The general sidebar tab has a few features:

- Cell size manipulation - Adjusting these allows users to change the scale (size) of cells displayed on the screen, the units here are ’px’ and can range from 1 to 999px.
- The Select All button will select or deselect all heat maps currently shown, this allows users to quickly manipulate only the data they wish too.
- The Close button simply closes / removes all selected heat maps from the screen and from memory.
- Parent and Child buttons allow the navigation through the history of all selected heat maps, another short-hand feature.
- Flip Y is a feature tat will invert the y axis of all selected heat maps. This feature can be useful when using the parallel coordinate overlay feature described later in this section.

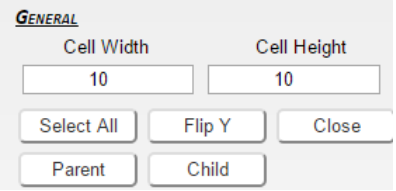


Figure 14: General sidebar tab

Tooltips To ensure the usability of the tool each user function has a tooltip which describes the action or parameters required by the tool. For example in figure 15 when a user hovers their mouse over the HTML spinner they are presented with a description on what the specific feature does. This kind of feature is good for usability as it does not impede the usability for advanced users and increases the usability for novice and beginner users as there is no need for the user to read full documentation and the information is presented while a user hesitates or is even looking for hints.

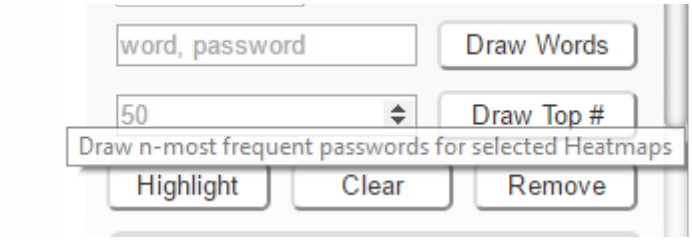


Figure 15: Example UI feature tooltip

Shortcuts Shortcuts are a feature aimed at advanced users, who already use keyboard shortcuts to increase their workflow. The tool implements only a couple keyboard shortcuts, however there is scope for many others to become available. Table 4 shows the keyboard combinations and there function. These were implemented while testing prototypes of the application and it became an annoyance having to select and deselect each heat map individually. The solution was to implement the standardised 'ctrl+a' to select and deselect all heat maps. The only other shortcut currently implemented is the delete (or close) function which, when triggered, removes all selected heatmaps from the screen, and memory. This can be done by pressing 'del' while at least one heat map is selected.

Key combination	Description
Ctrl + A	De-/Selects heat maps
del	Closes all selected heat maps

Table 4: Table of functional shortcuts

6.3.2 Visual customisation

One of the most important features for this tool with regard to usability was to enable users to customise as much as possible. All customisations are listed in Table 5, which shows each parameter available for customisation, a description of feature, and its default value. All of the customisations are regarding the visuals of the heat maps.

A user can modify these values in two ways: Firstly the user may modify the JSON object named userConfig.js (see Appendix 1 on page 56) which will permanently modify the visuals of the application. Otherwise these values are modified by adjusting the visual features in the configuration tab, and in the sidebar tab. These are however soft changes will not be preserved from session to session.

Note: kh, hh, hw, sbh, hish, and fs are variables that can only be modified by directly changing the JSON file. This is due to the fact that very little use comes from it and would require a separate configuration window, which was not a priority.

Note: Some values must exist as keys in the userOption.js object (see Appendix 2 on page 56 and Table 6 on the next page)

Key	Name	Default Value	Description
cw	Cell width	10	Width of heat map cells
ch	Cell height	10	Height of heat map cells
kh	Key height	15	Height of heat map key/legend
hh	Header height	15	Height of heat map X-axis label
hw	Header width	15	Width of Y-axis label
sbh	Strength Bar height	25	Height of the strength bar indicator
hish	Histogram height	25	Height of heat map histogram
fs	Font size	12	heat map font size
flipy	Flip Y	false	Flip the Y-axis of heat maps
maxLength	Max length	15	Maximum value of Y-axis (character position)
charSet	Character Set	“alphaNum”	Character set*
colourSet	Colour Set	“YlGn”	Colour set used to colour heat maps*
impossible-Cells	Impossible Cells	“grey”	Colour use to colour impossible cells*
colourMode	Colour Mode	“inverseAlpha”	Colour distribution function*
alpha	Alpha	2.5	Colour distribution function factor
axisOrder	Axis Order	“alphabetical”	Axis order*

* - Value must match a corresponding key in the user options JSON object.

Table 5: Configurable visual features

6.3.3 User configurable options

Coupled with the user configuration file (userConfig.js 1) is the user options file (see Appendix 2 on page 56). This is also a JSON formatted object and holds all of the options that the tool uses to set-up the user input fields used to modify the visualisations. Building on from the work done by Yin Lee [19], his functions have been reimplemented and improved for maintainability which will be discussed later.

Table 6 summarises the current feature options and gives the reader the format expected for modification with an example. The reader should also see Appendix 2 on page 56 to see the actual user options object. Any modifications made will be updated on start-up or on reload of the tool.

The format used is JSON and in general takes the form:

```
objectName = {
    key : value,
    key : [value1, value2, value3],
    key : {
        key : value4,
    }
}
```

Note: See the notes on Table 2 regarding some variables requiring special attention as the keys and values used may be specific to a function within the application.

Key	Name	Value format	Example
charSet^	Character sets	“charSet” : { ”key” : “string”, ..., },	”charSet” : { ”lowAlpha” : ”abcdef...”, ”highAlhpa” : ”ABCDE...”, },
colourSet^	Colour Set	“colourSet” : { ”key” : array[5], ..., },	”colourSet” : { ”Reds” : [”#fee5d9”, ...], ”Greys” : [”#f7f7f7”, ...], },
impossible-Cells^	Impossible Cells	“impossi...” : { ”key” : “string”, ... , },	”impossibleCells” : { ”grey” : ”Grey”, ”white” : ”White”, },
colourMode^*	Colour distribution mode	“colourMode” : { ”key” : “string”, ..., },	”colourMode” : { ”linear” : ”Linear”, ”alpha” : ”x^alpha”, },
axisOrder^*	Axis order	“axisOrder” : { “key” : “string”, ..., },	“axisOrder” : { “frequency” : “Frequency”, ”alphabetical” : ”Alphabetical”, },
alpha^	Colour distribution parameter	”alpha” : { ”min” : float, ”max” : float, ”step” : float, },	”alpha” : { ”min” : 1, ”max” : 10, ”step” : 0.1, },
maxLength^	Y axis max value	“maxLength” : { “min” : int, “max” : int, “step” : float, },	“maxLength” : { “min” : -1, “max” : 60, “step” : 1, },
flipy	Flip Y axis	“flipy” : boolean,	“flipy” : true,
debug	Debug flag	“debug” : boolean,	“debug” : true,

^ - These JSON objects are used to populate corresponding UI elements. The key is the name use, and the value is the value used.
* - These JSON objects value’s must exist in the source code as functions and appropriate modifications must be made, i.e., “alphabetical” is a function that sorts and returns the character set alphabetically (see heat map.js in Appendix 6 on page 61).

Table 6: Configurable user options

6.3.4 Filters

Several filters have been implemented for this tool with the aim to manipulate data in a user friendly way so that data can be teased out. The functions implemented are only basic functions and are actually rather limited in their usability, but they were required to show the tools overall usability. It is expected that this project will be continued in the future and advanced users to plug-in their own functions which makes use of the modular design structure of the source code. A user interacts with the filters from the sidebar, under the heading "filters" as shown in figure 16, and the filters operate on all selected heat maps.

See heat map.js in Appendix 6 on page 61 for the implemented methods regarding filtering.

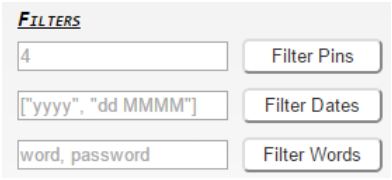


Figure 16: Filters side bar interface

PIN filter The PIN filter function filters for only numeric passwords of a given length (default 4). The function works by first getting all the words of length n and then checks each character in the password from left to right. At any instance where a character is not numeric then the word is discarded, otherwise the password is kept. The filtered list of passwords, of length n, and of numeric characters, is passed into a new instance of a heat map object which is given to a new instance of a heatmapUI object.

Date filter The date filter uses the library date.js and provides powerful functions for date processing. This project only makes use of the parseExact(string, [dfmt, dfmt, dfmt, ...]) function which takes an array of date formatted queries (i.e. MM = digit month, dd = digit day, yyyy = 4 digit formatted year, etc...) all of which can be found in their documentation [6]. The function will check if a password matches exactly each date format, it only need match one to return the password which is then used to create a child heat map.

The implemented function, again, is not too powerful and relies mostly on the user inputting the correct commands to tease out information, however due to time constraints it was seen only necessary to show what could possibly be shown using our visualisation method when viewing different dates.

Word Filter The word filter allows for the filtering of any number of strings. The implementation takes a comma separated string, converts that into an array of words to filter for, i.e., "password, 123456, 111111" is interpreted as filter("password" AND "123456" AND "111111"). This is a powerful method of filtering which allows for a direct comparison of semantics between heat maps (and therefore password sets).

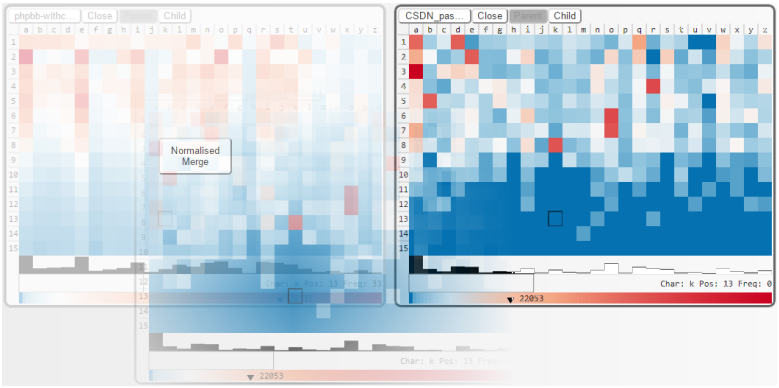
This filter works by firstly processing the user given string to an array of strings. Each filter word is then compared against every word in the heat map, if the filter word is contained within the password it is then kept, otherwise thrown. This is done for all selected heatmaps.

[6] http://www.datejs.com/

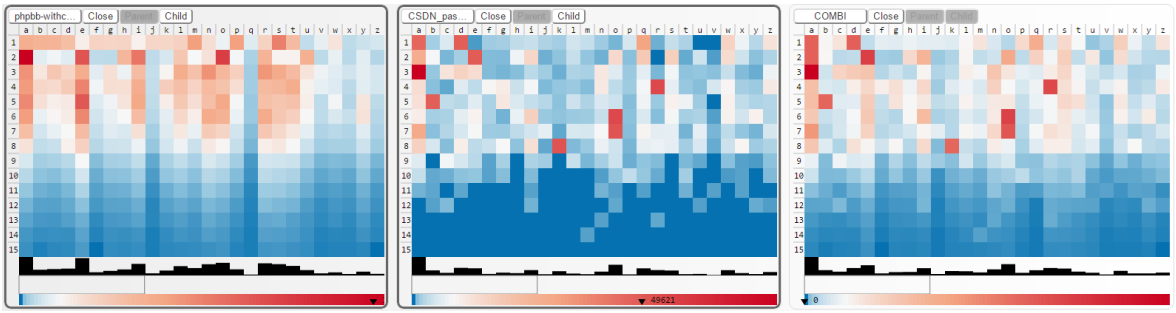
6.3.5 Drag & Drop heat maps

The drag and drop functionality allows users to intuitively combine heat maps. On drag start the selected heat map is shown under the cursor to indicate that this is able to be dragged. When entering a heat map while dragging a different heat map a notification screen is shown over the target, with an area titled “Normalised Merge” as shown in figure 6.3.5. On drop of the source heat map over the area titled normalised merge the program combines the heat maps and new heat map is created titled “COMBI” as is shown in figure 17b, more about the normalised merge feature in the next sub-subsection.

The purpose of this feature is that a user no longer needs to manually sort, or filter though their data and combine them using a text editor, this is especially useful as heat maps can be combined at any stage in their history, and abstracts the whole process into a user friendly way.



(a) Drag & drop interaction showing user notification drag-over



(b) Original heat maps (left & middle) v.s. Combined normalised heat map (right)

Figure 17: Drag & Drop interactions

6.3.6 Normalised dataset merge

The normalisation feature in this project allows for the merging of two heat maps. Each heat map object contains a method for merging and it takes a source heat map as its parameter. The method takes all words contained in the heat maps, normalises the source frequency against the target frequency, and creates a new heat map object with the two lists merged. See figure 17b for an example of two heat maps being merged. The left and middle heat maps have been merged to create the one on the right, named as “COMBI”. The reader should notice the difference and similarities between the three visualisations.

6.3.7 Cross Correlation Coefficient

Cross Correlation is a method to compute similarity between data, such as images, that have 2 dimensions (position and colour, or in our case position and frequency). The method produces a coefficient whose value is between 0 and 1 and represents the similarity between two sets of data, 1 being the same data/image.

The implemented function is developed form MATLAB's corr2 function which is documented in their software. The algorithm used is shown in figure 18. Our implementation slightly differs in that the data must be normalised before the cross correlation coefficient can be computed, once normalised the function computes the mean of both sets of data and follows the algorithm in figure 18.

corr2 computes the correlation coefficient using

$$r = \frac{\sum_m \sum_n (A_{mn} - \bar{A})(B_{mn} - \bar{B})}{\sqrt{\left(\sum_m \sum_n (A_{mn} - \bar{A})^2\right)\left(\sum_m \sum_n (B_{mn} - \bar{B})^2\right)}}$$

where $\bar{A}$ = mean2(A), and $\bar{B}$ = mean2(B).

Figure 18: MATLAB's Corr2 Algorithm

To calculate the corr2 function against heat maps a user only needs to select at least 2 heat maps for the results to show, which are displayed in the sidebar under the heading "Cross Correlation Coef". A coefficient value is computed between each selected heat map and is not sorted. Any number of heat maps may be compared simultaneously. A sample of results for the reader to review can be see in the sub-figures under figure 19, the reader should note that these results are from three data sets (phpbb, muslimMatch, and CSDN) against English, French, and German dictionaries in order to determine the language of the data sets, where it is clear phpbb and muslimMatch are closest matching to English and the CSDN doesn't match well with any of the languages due to the fact this is a mostly Chinese data set.

Cross Correlation Coef

```
0.5725 - phpbb-withco vs CSDN_passwor
0.9419 - phpbb-withco vs muslimMatch-
0.8400 - phpbb-withco vs french.dic
0.5418 - CSDN_passwor vs muslimMatch-
0.4673 - CSDN_passwor vs french.dic
0.7631 - muslimMatch- vs french.dic
```

Cross Correlation Coef

```
0.5725 - phpbb-withco vs CSDN_passwor
0.9419 - phpbb-withco vs muslimMatch-
0.7584 - phpbb-withco vs german.txt
0.5418 - CSDN_passwor vs muslimMatch-
0.4182 - CSDN_passwor vs german.txt
0.6758 - muslimMatch- vs german.txt
```

Cross Correlation Coef

```
0.5725 - phpbb-withco vs CSDN_passwor
0.9419 - phpbb-withco vs muslimMatch-
0.9190 - phpbb-withco vs english.txt
0.5418 - CSDN_passwor vs muslimMatch-
0.5300 - CSDN_passwor vs english.txt
0.8377 - muslimMatch- vs english.txt
```

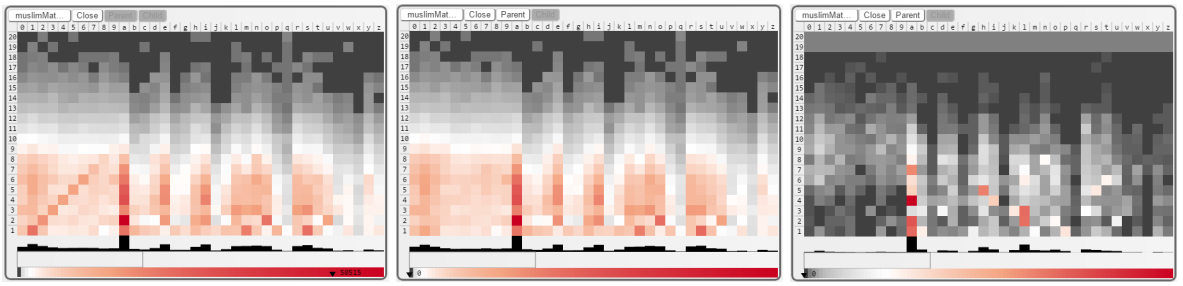
(a) Password lists v.s. French dictionary (b) Password lists v.s. German dictionary (c) Password lists v.s. English dictionary

Figure 19: Determining similarity of password lists against language dictionaries using Corr2

6.3.8 Heat map history

Data history is implemented in this software and is referred as Parent and Child. Each heat map has a reference to a parent heat map an a child heat map. Where a heat map has no parent it is referred to as the Root data set. A child is shown and created any time that the user manipulates the data, such as

filtering or removing passwords. Figures a, b, and c of figure 20 represent a chain of user interactions with the data, figure 20a shows the original untouched passwords set from the muslim-match list. figure 20b shows the data after the removal of the top 500 occurring passwords (notice the removal of the line “123456789” in the bottom left), the “parent” button is now enabled. In the last figure 20c the visualisation represents the 1st child after filtering for words that include “allah”, “god”, “mekka”, “bar”, “paki”, “muh”. It is now possible for the user to navigate back and fourth between the three sets of data which allows easy comparison.



(a) Raw Muslim match dataset (b) 1st child - Remove top 500 occurring passwords (c) 2nd child - Words including allah, god, mekka, bar, paki, muh

Figure 20: Parent/Child/History interaction

6.3.9 Heat map statistics

Statistical information about the imported datasets is important to get a grip on the range of data the user is looking at. The sidebar contains a separate section for the stats, and only shows the stats of those heat maps that are selected. A comparison of key statistics can be made as well as helping the user gather further meaning about the data they are reviewing. See figure 21. The stats computed are as followed:

- File Name: File name of the imported data.
- Word Count: Total words represented by the data (sum of frequencies).
- Char Count: Total characters represented by the data (sum of each word length multiplied by its frequency).
- Max Length: The length of the longest password.
- Max Frequency: The frequency of the most frequent word.
- Unique Words: Total unique words in the data set.
- Character Frequency: An Interactive character histogram representing the frequency over the visible character set.

The character frequency chart will be explained further in the next sub-subsection.

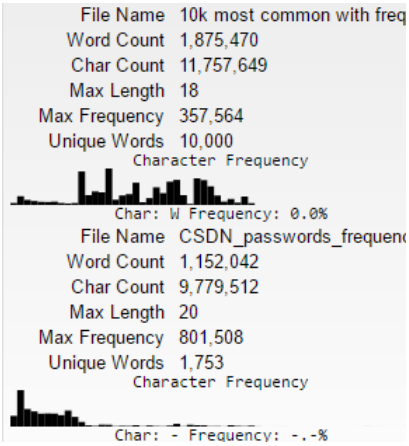


Figure 21: Statistics interface for CSDN and 10k top passwords with responsive histogram

6.3.10 Character histogram

Character histograms can appear twice, they are on every heat map with each bar under its representative character from the heat map, and again in the sidebar stats tab if it is selected. The reason for this is to allow users to quickly review the character distributions between the heat map and also between heat maps. The heat map implemented in the sidebar, shown in figure 21, are interactive to give more information to the user if they wish. Hovering over each bar update the key below of the character and frequency represented by that bar. The histogram implemented under each heat map is not interactive as it is easily inferred which character represents each bar. See figure 22.

This paper recommend that future implementations should change the heat map histogram from a bar chart to a heat map visualisation, where the X-axis labels have background colour of the relative frequency of that character. This is to help with consistency and readability.

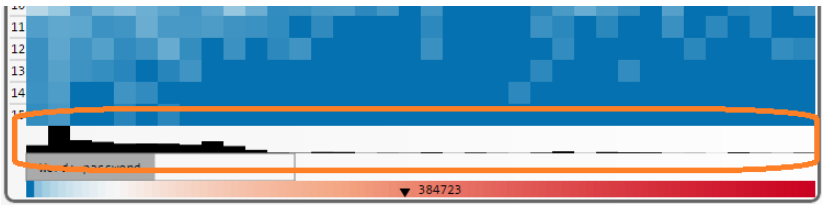


Figure 22: heat map character histogram

6.3.11 Multiple file import

Users must import files into the program so that they can be visualised. The user has the option of uploading one or many files at once, however the files should all be in the format that the program accepts. The following table 7 show three formats that are accepted by the current implementation:

Headers: **Data:**	password,frequency string,number ...,...	frequency,password number,string ...,...	password string ...

Table 7: Upload file format

Files are first uploaded and then processed by d3.js before a heat map object is build from the data. Failure to meet the above format will result in buggy behaviour and in the there are plans to implement an import files feature which should minimise bugs and increase the file formats accepted and therefore increase the usability of the tool.

While files are being uploaded via AJAX a splash screen is displayed to notify the user. This splash screen also prevents the tool from being used while the files are being added to the screen. See figure 23.

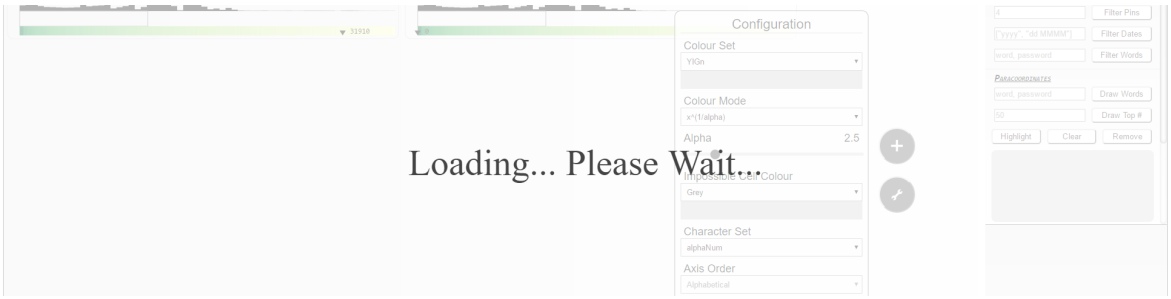


Figure 23: File upload splash screen

6.3.12 Synchronised UI

To help users compare data an attempt to synchronise features over all visualisations has been made.

- Synchronised cell selection
- Synchronised password selection
- Synchronised password strength meter

As a user hovers over cells the tool shows the character, position, and frequency at that point. This feature is demonstrated in figure 24 where the orange highlighted boxes show the synchronised behaviour and represent data from each graph. This feature words in real time and only requires a mouse pointer to hover over any cell.

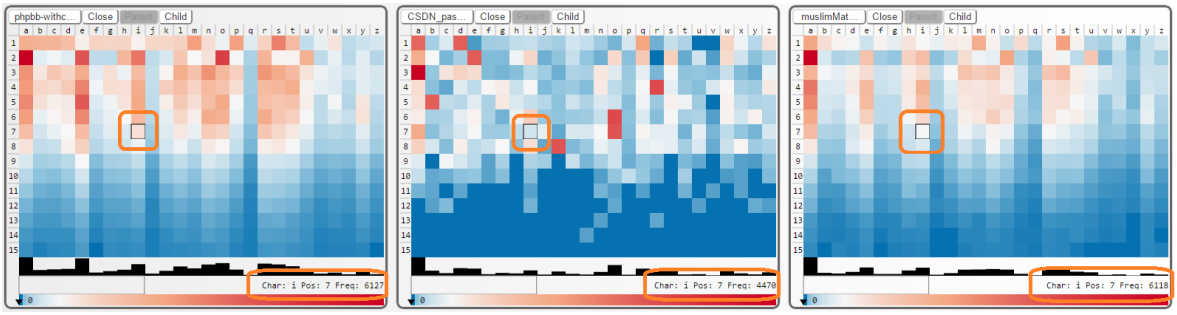


Figure 24: Synchronised cell tool tips across all heat maps

When visualising words using parallel coordinates over the heat map the lines will be highlighted underneath the cursor and display the password strength in the bottom left of the heat map window. This is a very useful feature for gathering an understanding of the data a user is looking at as well as evaluating the strength of a password across all selected heat maps. See figure 25. A user may also click any word to keep it selected, in this case the line is highlighted in red to increase its visibility.

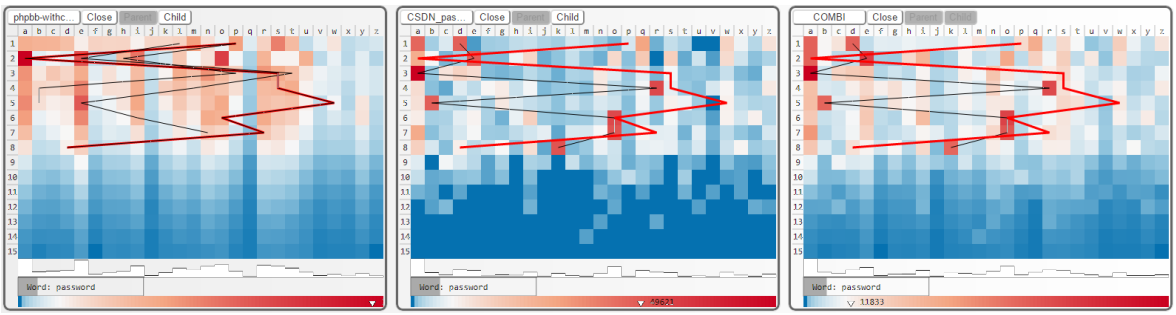


Figure 25: Synchronised line selection and password strength across all heat maps

6.3.13 Pop-out configuration pane

Configuration of the visual properties of the heat map and axis ordering is undertaken in the pop out configuration pane. This pane is translucent so that for small devices a user can review their choices without having to hide the pane. The pane can also be moved anywhere on the screen as well as being hidden at any time, which helps again with smaller devices which do not have as much screen space.

The pane allows the user to change, in real time, the colour set, colour mode, the colour mode scaling factor, colour of impossible cells, character set (x-axis), and the max length shown (y-axis). These fields are populated and configured by the userConfig.js and userOption.js files as mentioned earlier in this subsection. See Appendix 1 and 2. Other features implemented on the pane are the colour keys for the colour set, which is updated depending on the colour set, colour mode, and the value of alpha, and any adjustment of the range bars update a label showing the numerical value of the range bar is updated during drag.

One final feature implemented here are two buttons, seen top right of figure 26. These buttons are movable and translucent for the same reason as mentioned for the configuration pane. The plus button toggles the upload file interface and the spanner button toggles the configuration pane itself.

6.3.14 Graphical password checker

One of the key aims of this project was to find a way to help educate users and policy makers on password strength. The graphical password checker helps do this. The strength is of a password is determined by the average frequency of each letter in the word against each heat map. This provides a measure of the password frequency in each set however doesn't take full consideration into some important factors such as length of the word, common patterns used to crack passwords, and character sets used.

The figures 27a and 27b show a comparison of two different password against two different password sets. The password "123456" (illustrated in figure 27a) is given a lower score against the CSDN list versus the phpbb list. This is due to the higher frequency of numeric passwords in the CSDN password list and the fact that the CSDN list is comprised of Chinese passwords. The password "123456" should however be given a overall much lower score than given in both cases due to the fact it would be one of the first passwords a password cracker would try due to the known frequency that users user sequential numbers for passwords. In figure 27b the strength of the word "!P4SS*RD+" is plotted, and rightly so gets a much higher score on the meter when compared to the strength of "123456", but the reader should still notice the still quite large difference in strength between the two password lists.

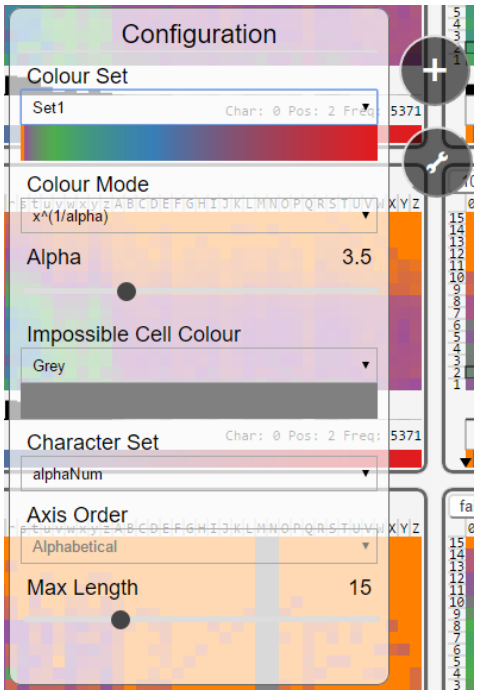
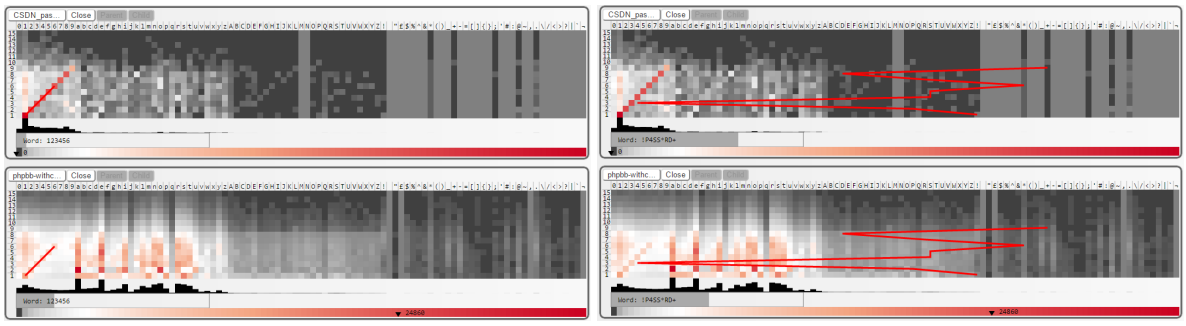


Figure 26: Pop-out configuration pane with upload button and configuration toggle

An additional feature to help with the visualisation of the strength of a password is to draw the line of the password, which is also highlighted to differentiate this line from other parallel coordinate lines that may be already drawn on the heat maps. This visualisation of the word couples with the graphical strength meter allows the user to further evaluate what made their chosen password weak and what character they could add to strengthen the password. In the future it would seem natural and give suggestions to increase the strength of a password, however this would need to be done in a non-predictable way which may prove to be difficult.



(a) Strength of word “123456” (b) Strength of word “!P4SS*RD+”

Figure 27: Graphical password strength meter

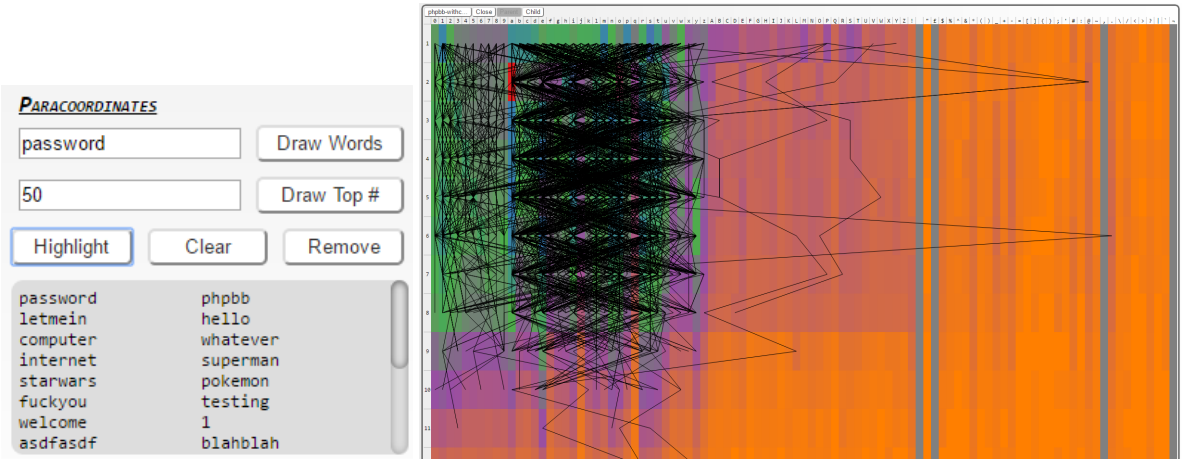
6.3.15 Parallel coordinate overlay

A key feature of this tool in terms of visualisation is the combination of parallel coordinates to visualise words and the use of heat maps to visualise the character patterns of passwords. The controls for parallel coordinates is in the sidebar under the heading “Paracoordinates” and gives the user some

functions the addition and removal of word lines as well as displaying the actual words represented by the tool, displayed in order of frequency, and allows for the removal of any or all selected word lines from the data, creating a child heat map. See figure 28a. Parallel lines are also highlighted and can be highlighted either by clicking them or hovering over the lines, doing so colours them red, and the word is shown in plaintext in the strength bar indicator also showing the passwords strength (as discussed in an above sub-subsection). The sidebar interface for parallel coordinates has the following features:

- Draw words - This feature draws all comma separated strings in the text field adjacent to the button on each selected heat map. See figure 29.
- Draw top # - Draws the top-n frequent passwords over each selected heat map. See figure 28b.
- Highlight - Highlights all parallel coordinate lines on the selected heat maps and displays the passwords in container below the button shown at the bottom of figure 28a.
- Clear - Clear all lines from all selected heat maps.
- Remove - Removes all selected word lines from each selected heat map.

The ability to remove the top-n words allows a user to explore the finer details in a password set, this is because the brightness cause by the most frequent passwords may drown out subtleties in the data.



(a) Parallel coordinate side bar interface (b) Visualisation of the top 1000 occurring passwords in the set

Figure 28: Parallel Coordinate sidebar and Visualisation

6.3.16 Flexible & fixed display

The tool has been developed with the ability to change the size of visualisations so that visualisations could be compared on the same screen. In figure 30 the reader is able to compare the 6 different heat maps simultaneously, picking up on patterns present or absent in each. The display of each UI component has a fixed size and the overall size of the tool is limited by the size of the device's window, where the internal components have been given a percentage of the window. This allows for the sidebar to be a fixed entity where the user options and statistical panels share the height of the

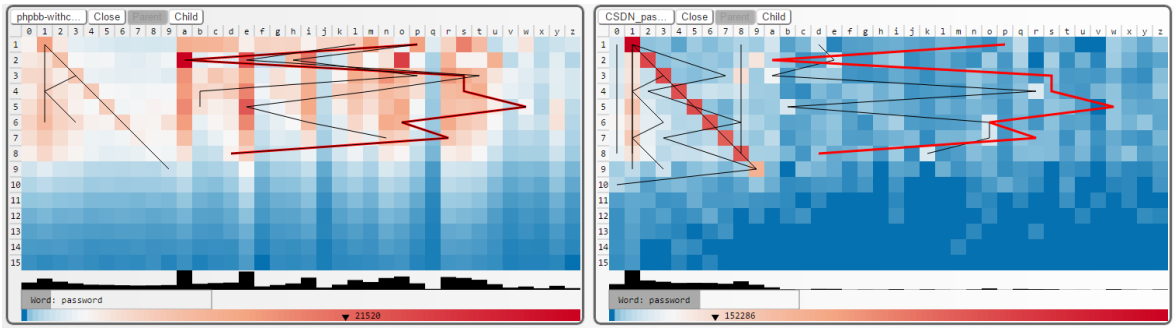


Figure 29: Drawing user defined words over the heat map

device, and no more. The rest of the screen size is made available for the visualisations, which are held within their own container, which results in the user being able to scroll though the visualisations while always being able to view the configuration pane and sidebar.

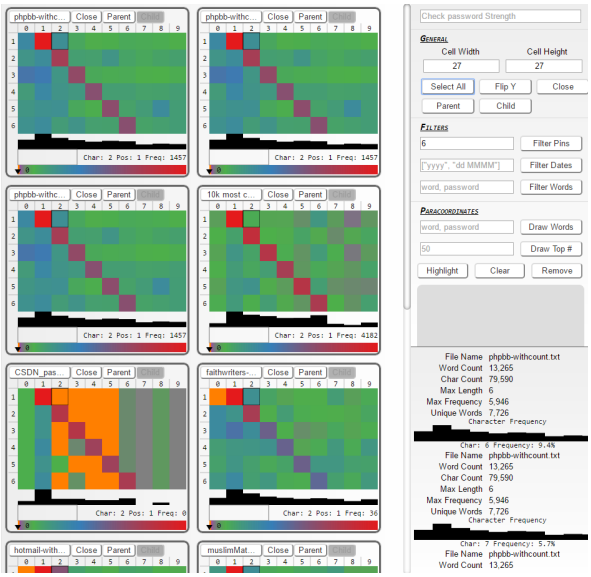
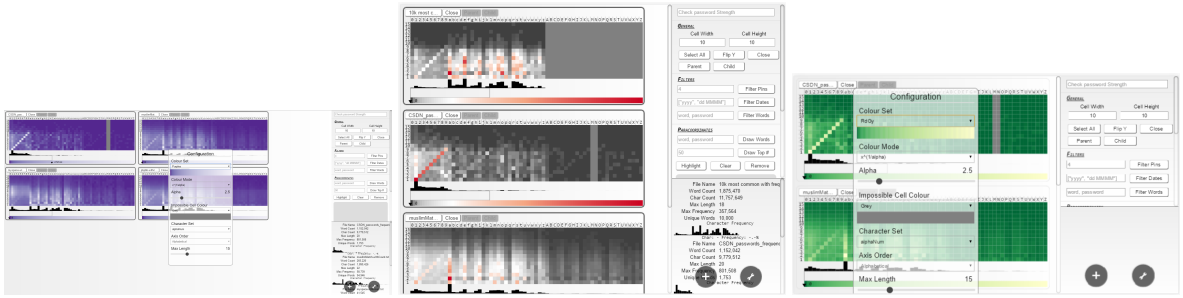


Figure 30: Display of multiple datasets for simultaneous comparison

In figure 31 the tool is tested in multiple environments, such as on an iPad, FHD desktop, and on a Nexus 6P mobile phone. In each it is possible to use the application to perform all functions, with exception to some touch specific functions on touch screen devices such as drag and drop. This is however only an issue of implementing minor changes to allow for mobile/touch interactions.

6.4 Summary

In summary the tool contains quite a bit of functionality that doesn't actually seem too apparent when first using the tool, which is a good thing. The tool manages to pack a lot of functionality on the screen without too much cluttering while also maintaining the core focus of the tool which is to visualise passwords to enable pattern discovery by comparing, filtering, and otherwise manipulating lists of passwords, words, etc. Compared to the original design of the tool the implementation has diverged some what due to time constraints and problem complexity, and these are discussed further the conclusions section.



(a) Desktop (1080p) (b) iPad (landscape) (c) Nexus 6P (landscape)

Figure 31: View of tool on multiple devices

7 Testing and Results

In this section we discover whether or not the final implemented system meets the requirements outlined during the requirements gathering process from users and the lessons learned from prototypes. The tests performed are aimed to show that the software also meets the aims of the project outlines in the introduction section. A successful fulfilment of the requirements and the aims of the project would mean a successful project.

7.1 Functional Testing

Testing of the functional requirements assigned for the system helps evaluate the successfulness of the system. A user performs a a task to test each functional requirement, the results and comments can be reviewed in the table below. The overall outcome for these tests suggest that there needs to be more work done to meet all of the requirements, however most requirements are met.

No.	Test	Expected Outcome	Actual Outcome
1	User attempts to manipulate data.	User should be able to manipulate the data intuitively.	Partial success. Functions for manipulating data require strict formulation, however some functions allow for intuitive manipulation such as the drag and drop feature.
2	User attempts to visualise leaked password data.	User should be able to visualise lists of passwords as well as individual passwords.	Success. The system allows for multiple visualisations, and for the visualisation of individual by overlaying techniques.
3	User attempts to customise the visualisations.	User should be able to customise settings via userConfig.js and userOptions.js. As well as run time customisation of the visualisations.	Success. User can customise visualisations interactively using the configuration pane and general sidebar tab. With some prior knowledge the user is able to change program defaults and configuration parameters by adjusting the userConfig and userOption JSON files.

4	User attempts to interact with the visualisations in real time.	User should expect to interact with visualisations in real time.	Success. User is able to drag and drop , select and manipulate specific visualisations. User can click, and is notified with mouse-over events.
5	User attempts to import multiple files.	User should be able to import multiple files.	Success. User is able to import an arbitrary number of files and are notified of the system status.
6	User attempts to import multiple formats of password lists.	User should be able to import varying types of formatted password lists.	Failed. Only CSV format is supported, though data visualisation is still attempted without errors, visualisations are wrong.
7	User attempts long processes with multiple inputs.	User should be informed of system status at any time. The system should remain responsive.	Failed. Some functions performed seem to hang the browser, and take too long, without notifying the user of system status.
8	User attempts to compare visualisations.	User should expect to compare the visual similarity of visualisations, as well as a statistical measure of similarity.	Success. User can view heat maps side by side, as well as combine data. Visualisations are also compared using the cross correlation coefficient in the CCC sidebar tab.
9	User attempts to view statistical information about a selected visualisation.	User should be able to review the statistical information about each selected visualisation of the data.	Success. User can review the statistical information in the sidebar under the statistics tab.
10	User attempts to visualise the strength of a password.	User should be able test the strength of a password with an accompanying visualisation.	Success. User can visualise the strength of their password over the visualisation of the data.

7.2 Usability Testing

Usability testing is used to evaluate the usability of the system for real users. The usability for this application is important as we wish for people to use this tool to find patterns easily, we wish not for users to spend too much time learning to use the tool. Usability tests were conducted by the Author by reviewing a user trying to use the tool, and an evaluation against Nielson's heuristics.

While monitoring a user using the application is was clear there were some minor usability issues for a new user. The main being that it was difficult for a new user to understand that this system is meant to be used for, however once introduced to the application and its functions, by way of tool tips and some Q&A, a user is able to start using the program to visualise data quickly and start exploring

the different functions.

In addition to a user review, the author reviews the system against Nielsen's heuristics, the table below showing the results. The results show that most usability heuristics are met, however there are improvements to be made.

No.	Heuristic	Result	Comments
1	Visibility of system status.	Partially Achieved	User is not notified about some long processes.
2	Match between system and the real world.	Achieved	Language is used naturally and appropriately.
3	User control and freedom.	Achieved	View of visualisations and configuration pane is flexible.
4	Consistency and standards.	Achieved	Each visualisation can be customised, however on first load will all have the same view.
5	Error prevention.	Failed	Error prevention is not fully implemented.
6	Recognition rather than recall.	Achieved	Novice users
7	Flexibility and efficiency of use.	Achieved	A user can customise the visualisations.
8	Aesthetic and minimalist design.	Achieved	The interface is clean and spacious and intuitive interactions are implemented.
9	Help users recognize, diagnose, and recover from errors.	Partially Achieved	More needs to be done to prevent errors as well help users recover. Visualisation history is implemented to allow users to recover from mistakes.
10	Help and documentation.	Achieved	This report plus tool tips help document the features of the system.

7.3 Stress Testing

Stress testing evaluates the stability of the system under varying circumstances which aim to exceed the expected usage that the application is designed for. Manual testing is done for this process due the specific format required by some of the features implemented, such as date filtering. The following observations were made.

- The word and date filters perform well for small sets of words, say less than 500 variables. When this limit is exceeded the program browser can sometimes hang, and the operation begins to take an unreasonable amount of time to complete.

- The Google Chrome web browser has an upper limit on the memory a tab can use. Due to the implementation files are loaded directly into the main threads memory, which means that there is an upper limit on the amount of data that can be shown by the application. The tests generally have shown that it is possible to view and process approximately 50MB, which is roughly 3 million passwords.

- Older devices and mobile devices often take longer to process queries such as filtering, file importing, and history viewing.

7.4 Compatibility Testing

It is important that the system works on many systems, and requires little preparation to be run so that the tool can be used by as many people as possible, and increase the chance that people want to use this tool. The system has been designed to work with web browsers, and so it is expected that it runs on most devices which have a web browser.

Tests conducted on Google Chrome, FireFox and Microsoft Edge are conducted. The tool and all of it's functions work as expected on all browsers on desktop devices. For mobile devices the tool has limited interactive functionality, but all other functions work such as the visualisation and filtering aspects. More work needs to be done to allow for drag and drop events in mobile browser implementations for this system. It should be noted that this system has not been tested with the Safari browser by Apple, though it is expect that this system will work as the Safari browser implements all of the web standards used by this project.

7.5 Maintainability Testing

Code metrics are used to give a measure to the maintainability of an application by attempting to combine features such as lines of code, functions, and nested functions. The results for the three source files heatmap.js, heatmapUI.js, and paracoord.js are shown in table 10. The total line count is quite low, which is good. The mean parameter count is high for heatmapUI however this is because it contains many variables conserved with UI components. It could be said that more should be done to separate this class more to improve its maintainability. The average Cyclomatic complexity density is 13%, which is good. This value describes the amount of self referencing and complexity of nesting functions, where less is seen as more maintainable. The final value used in this metrics test is the maintainability index. this value resolves all the other metrics and computes the overall maintainability of the code, the higher the value the better. All three classes perform well here scoring over 100.

	heatmap.js	heatmapUI.js	paracoord.js
LOC	224	434	649
Mean parameter count	21	48	29
Cyclomatic complexity	34	47	95
Cyclomatic complexity density	15%	11%	15%
Maintainability index	105	106	108

Table 10: Results of code analysis

7.6 Results and Observations

The final test for this project is to evaluate the systems primary function, which is to help with the visualisation of known and unknown patterns. In this subsection a presentation of the results are shown in an attempt to meet the following aims:

- Visualise known password patterns better.
- Visualise new password patterns.

In a comparison of 4 leaked password lists, muslim match, myspace, phpbb, and rockyou, we find that they are visually all very similar, sharing the same visual patterns on the heat map visualisations. This fact is also backed up by the cross correlation coefficient shown below. It was expected that the rockyou dataset would score highly with the other due to the huge sample size represented by it (7 million users)

	Coefficient
phpbb vs rockyou	0.9850
myspace vs rockyou	0.9589
myspace vs phpbb	0.9545
muslim match vs phpbb	0.9456
muslim match vs rockyou	0.9441
muslim match vs myspace	0.8830

The system can be used to determine some demographical information about the users in each password list. We can first compare visualisations against language dictionaries to determine the main languages used by an organisations user base. Comparing the 4 datasets (muslim match, myspace, phpbb, and rockyou) it is concluded that their highest user demographic is that from English speaking countries It is interesting to note here that muslim match has the lowest score of English, French and German, which would make sense as many Muslims, who use muslim match, would speak some kind of Arabic dialect. The CCC results are below.

	muslim match	myspace	phpbb	rockyou
English	0.8377	0.8997	0.9190	0.9007
French	0.7631	0.8207	0.8400	0.8254
German	0.6758	0.7396	0.7584	0.7422

Comparing now the digits present in passwords between the 4 sets of passwords it is clear there are a few patterns shared by all, with some exception to the myspace data set, where passwords appear to not contain numbers in the middle, i.e. pa55word, or whole number passwords such as 123456, see figure 32. The other sets of passwords clearly show a strong pattern for runs of digits (123456), and in the top left of each visualisation a high use of numbers is obvious. This would be due to the fact many users end there passwords with a number. However the use of numbers at the end of passwords is not uniformly spread, such that only digits 1, 2, and 3 are used more frequently that others. The very strong use of 1 will be due to users picking year's, such as 1???. Another reason for this high use of 1 could be due to the fact users like to end their password with the number 1. Which has been noticed as a common theme in passwords. The number 1 at the beginning of a password is also highly common, which again has been shown in previous research.

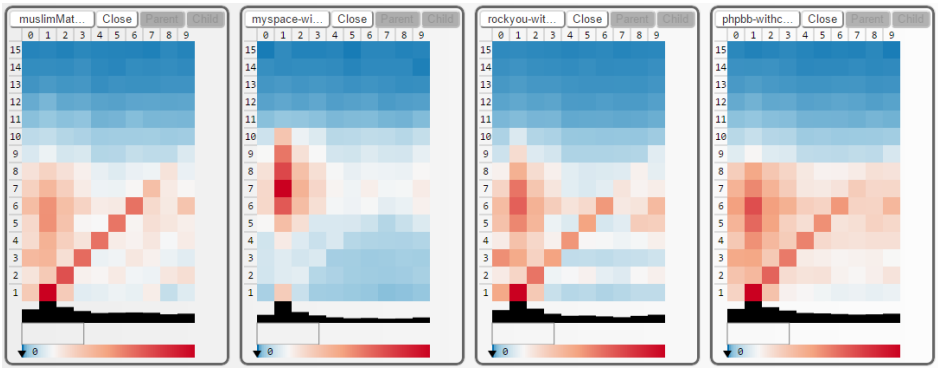


Figure 32: Patterns in digits

We notice some unknown pattern in the phpbb dataset. See figure 33. There seems to be some kind of pattern across the low and high alphabet. The reader should see the positive slope from b to e, g to i, k to m, as well as the negative slopes from n to q, and s to v. Now some of this can be explained by the frequency character distribution of the English language, which is the dominate language as mentioned earlier. However it does not quite explain it all. The author has a suspicion that these patterns might be further explained by further visualisation of the data to show keyboard patterns as there may be some relationships to do with how close keys are on the keyboard, such as qwerty, or common patterns such as abc.

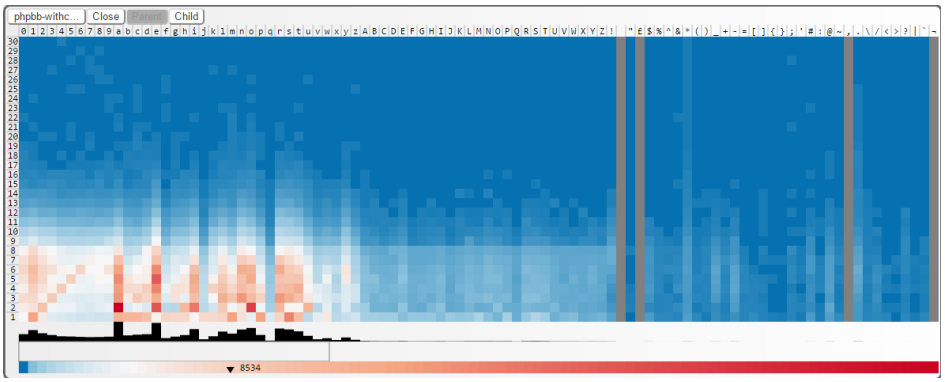


Figure 33: Unknown patterns

Using the parallel coordinates visualisation it is possible to see some common traits used in passwords, see figure ##. The patterns present here show that the myspace passwords are mostly low alpha followed by a number. This trait is not shared across all password datasets.

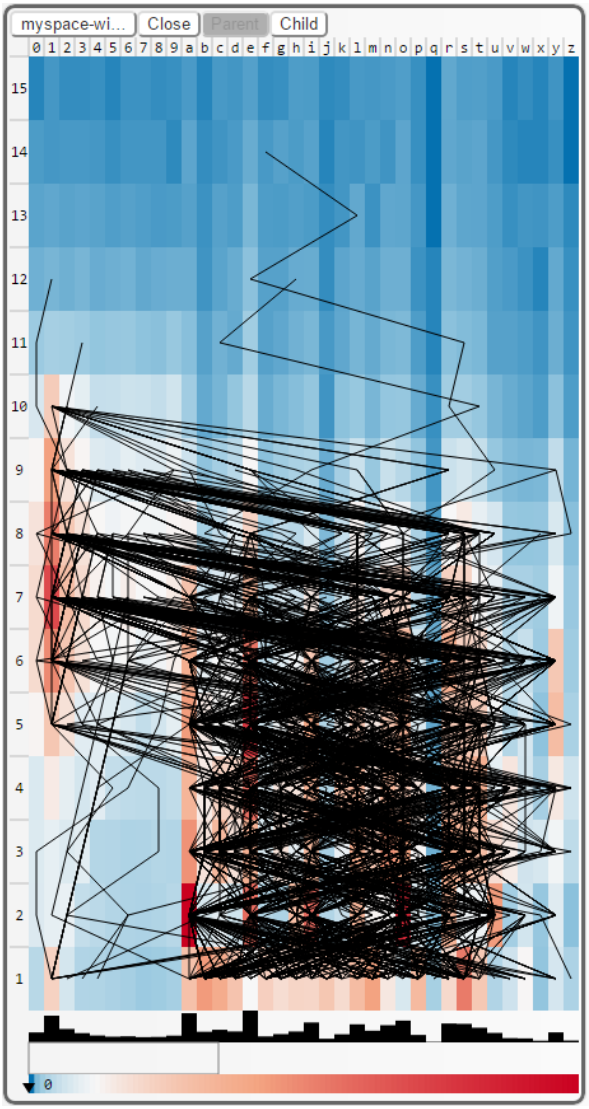


Figure 34: Alpha followed by number pattern in myspace

One last discovery that the author found was that using this tool can further explore the demographics for each set of passwords. Filtering for a specific semantic allowed for more information to be learnt about the users. It was found that muslim match contained a much high proportion of users using the passwords referring to words to do with the Muslim religion (see figure 35), which is expected as we knew the source of the data. The author is is sure that further searches such as this will result in more results of significance.

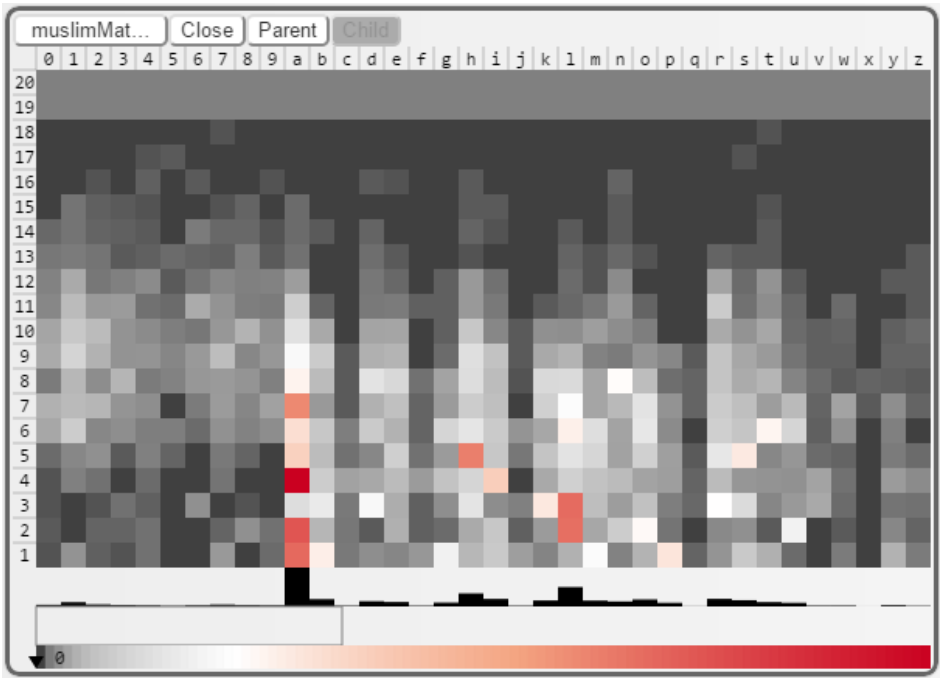


Figure 35: Filtering muslim match for words including allah, god, mekka, bar, paki, muh

7.7 Summary

Overall we see that it has been shown that a functional system has been produced, fulfilling many of the aims and requirements of the project. However it is clear to the author that more work is required for the implementation for more complex features that were not possible to implement here due to tie constraints and the scope of the project, such that it would have been useful for the system to allow for some kind of automatic semantic detection of passwords, however it is discovered during the literature review that this task is non-trivial and has many complications worthy of a project all of its own. Therefore to summarise the tests it was shown that the system met the must have requirements, and some others. The maintainable of the system scores well, and the usability of the system is shown to meet most of Nielsen's usability heuristics though more work can be done here to improve the usability.

8 Conclusion

This final sections concludes this report. Here a evaluation of the project as a whole is conducted, taking into account the aims, objects, and requirements outlined earlier in this project. The goal here is to access how well the project has performed in respect to what the author envisioned. A discussion of the difficulties faced during the production of the system and this report. The author also discusses future work possible from this point to help any project that continue on from here. Finally this section concludes with a personal reflection

8.1 Project Evaluation

Given the time frame available the project has been completed, though there is work to done still which is mentioned in the following subsection. The system developed has been shown to view patterns in passwords and has enabled us to find some interesting characteristics about sets of leaked passwords. The interactivity of the visualisations help improve on previous work done on the visualisation of passwords done by Yin Lee.

We started this project with a review of literature to understand what is known about passwords, how they are cracked, and what is being done to further our knowledge of passwords such as discovering trends in human behaviour. Current visualisation techniques used to visualise passwords and their patterns are explored and it is discovered that little work has been done, allowing for this project to be nicely positioned to perform well. Following the review of materials and methods we discuss the requirements and create concepts based on these. Several high level ideas are discusses during the design phase however during the implementation phase it is found that some ideas were not feasible given the time frame and expertise from the author. These ideas are further discusses in the future work section. Despite not all the functions being implemented from the design phase we have shown good results and meet our aims for the project.

Finally to evaluate the successfulness of this project we evaluate the objectives of the project again. Table 11 shows the objects and their state, now the project has been completed. We conclude that we have met all the objectives for this project and this conclude the project has been a success in terms of meeting the objectives.

Objective	State
Review known password patterns.	Achieved
Examine current methods in password visualisation.	Achieved
Review visualisation techniques.	Achieved
Identify requirements.	Achieved
Identify and review multiple concepts based on requirements.	Achieved
Implement solutions based on conceptual designs.	Achieved
Test and evaluate the system.	Achieved

Table 11: Project objectives

8.2 Limitations and Future Work

Although all of the objectives of this project have been achieved and it would seem this project was a complete success, it really is not that clear. The final implemented system has its flaws and there

clearly needs to be more work done to implement features that will further help with the discovery of patterns. Feedback from Shujun, the prime user, suggested that more file formats should be supported, and he even suggests that a converter is implemented for the either manual or automatic conversion of password lists. In other feedback it was requested that the visualisations should be auto scaled to fit inside the available space. This feature could be further improved to allow users to customise the number of visualisations to appear per column or per row.

During the testing of the system, to evaluate its features, it became clear that further efforts should be made with the filtering. It is suggested that future work should aim to allow for the automatic semantic categorisation as shown by the work done by Veras in 2012 and further in 2014[6, 21]. To visualise these semantics better the reader should refer to the concepts discussed in section 5 relating to the parallel coordinate plots. Unfortunately the time line allocated for the implementation was not enough to attempt to implement these features into the system even though they are sure to improve useful pattern discovery. This visualisation technique coupled with the a more user friendly interface for filtering as that proposed in the design section for the implementation of a drop down menu10 which allows users to pick semantic categories rather than creating a list manually in an attempt to filter the data into semantic categories.

Other limitations of the system is the, already identified, graphical password checker. The passwords checker should in future take into account passwords entropy as well as the frequency of character in the dataset, and even could use the frequency of semantic categories as a feature of password strength, too. One other recommendation to be made here is that the password checker should attempt to suggest a better password given a weaker passwords as a seed. The final limitation is that there is a maximum amount of data that the browser can store in memory before it crashes, and this is unfortunately rather limiting as it turned out. In tests it was not possible to load the whole RockYou data set, consisting of over 7 million passwords and their frequencies. Some attempts were made to find work around are none were found, thus resorting to use only two-thirds of the RockYou data set in the tests. It is hoped that future work look at a fix for this.

8.3 Personal Reflection

The completion of this project now allows for time to reflect on the processes involved with the management, design, and development of the project and the system produced. In the beginning it was proposed that the goal of this project was to simply expand the ideas developed by Yin Lee and the heat map visualisation software for web browsers, later on, the project has stretched its goals and stretched the author as a developer and his project management. Together with Shujun, ideas were discussed and discussions progressed into implementations. This kind of development process is a particularly enjoyable one as it allows for the rapid creation and evaluation of features for the system. A feature would be implemented and soon after Shujun would comment with feedback as to whether or not this was useful, and with this adjustments would be made, or a note was made for future development if it was not an urgent feature or was too big to fit into this project.

Back in August I secured employment as a Software Engineer at Freelancer.co.uk, the development skills developed during the creation of this report and of the software have most definitely refreshed and improved my skills and project management. However in retrospect I should have spread the written aspect of project through the year rather than leaving until the last few months. The

pressure from the looming deadlines has however improved and thinned out my workflow, increasing productivity massively, though it is important to remember that life will not always allow for such a smooth timeline and I should not only rely on this skill to produce work at the last hour.

The research conducted during the literature review and during the requirements and analysis process has given me a wider breadth of knowledge, deepening my understanding of human behaviour, which is a fascinating topic, the visualisation of data, and with the security implications of passwords and the research being done to improve and to guess/crack passwords in as little guesses as possible. It certainly pleases me to know that this project has matched some of the literature's results and have found some other unknown patterns. So I am quite excited to continue working with Shujun after my time at Surrey in my spare time, with the aim to improve the tool further and get some good results which could become published research.

References

[1] C. H. Dinei Florencio, "A large scale study of web password habits," Tech. Rep., November 2006. [Online]. Available: https://www.microsoft.com/en-us/research/publication/a-large-scale-study-of-web-password-habits/

[2] D. Wheeler. (2012) zxcvbn: realistic password strength estimation. [Online]. Available: https://blogs.dropbox.com/tech/2012/04/zxcvbn-realistic-password-strength-estimation/

[3] M. Burnett. (2011, June) 10,000 top passwords. [Online]. Available: https://xato.net/10-000-top-passwords-6d6380716fe0#.jsxjxk89p

[4] A. Vance, D. Eargle, K. Ouimet, and D. Straub, "Enhancing password security through interactive fear appeals: A web-based field experiment," in *System Sciences (HICSS), 2013 46th Hawaii International Conference on*. IEEE, 2013.

[5] M. Dell'Amico, P. Michiardi, and Y. Roudier, "Password strength: An empirical analysis." in *INFOCOM*, vol. 10, 2010.

[6] R. Veras, C. Collins, and J. Thorpe, "On semantic patterns of passwords and their security impact." in *NDSS*, 2014.

[7] P. G. Inglesant and M. A. Sasse, "The true cost of unusable password policies: password use in the wild," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2010.

[8] J. Toon. (2013) Teraflop troubles: The power of graphics processing units may threaten the worlds password security system. [Online]. Available: http://www.gtri.gatech.edu/casestudy/Teraflop-Troubles-Power-Graphics-Processing-Units-GPUs-Password-Security-System/

[9] J. Bonneau, "The science of guessing: analyzing an anonymized corpus of 70 million passwords," in *2012 IEEE Symposium on Security and Privacy*. IEEE, 2012.

[10] N. Anderson. (2013) How i became a password cracker: Cracking passwords is officially a "script kiddie" activity now. [Online]. Available: http://arstechnica.com/security/2013/03/how-i-became-a-password-cracker/

[11] C. M. Weir, "Using probabilistic techniques to aid in password cracking attacks," 2016.

[12] M. Weir, S. Aggarwal, M. Collins, and H. Stern, "Testing metrics for password creation policies by attacking large sets of revealed passwords," in *Proceedings of the 17th ACM conference on Computer and communications security*. ACM, 2010.

[13] B. Ur, J. Bees, and Segreti, "Do users perceptions of password security match reality?" in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2016.

[14] B. Ur, F. Noma, J. Bees, S. M. Segreti, R. Shay, L. Bauer, N. Christin, and L. F. Cranor, "" i added'!'at the end to make it secure": Observing password creation in the lab," in *Eleventh Symposium On Usable Privacy and Security (SOUPS 2015)*, 2015.

[15] B. Ur, P. G. Kelley, S. Komanduri, J. Lee, M. Maass, M. L. Mazurek, T. Passaro, R. Shay, T. Vidas, L. Bauer *et al.*, "How does your password measure up? the effect of strength meters on password creation," in *Presented as part of the 21st USENIX Security Symposium (USENIX Security 12)*, 2012.

[16] R. Shay, S. Komanduri, P. G. Kelley, P. G. Leon, M. L. Mazurek, L. Bauer, N. Christin, and L. F. Cranor, "Encountering stronger password requirements: user attitudes and behaviors," in *Proceedings of the Sixth Symposium on Usable Privacy and Security*. ACM, 2010.

[17] M. L. Mazurek, S. Komanduri, T. Vidas, L. Bauer, N. Christin, L. F. Cranor, P. G. Kelley, R. Shay, and B. Ur, "Measuring password guessability for an entire university," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013.

[18] N. Berry. (2012) Pin number analysis. [Online]. Available: http://www.datagenetics.com/blog/september32012/

[19] Y. T. Lee, "Texture password visualisation," 2015.

[20] X. Guo, H. Chen, X. Liu, X. Xu, and Z. Chen, "The scale-free network of passwords: Visualization and estimation of empirical passwords," *arXiv preprint arXiv:1511.08324*, 2015.

[21] R. Veras, J. Thorpe, and C. Collins, "Visualizing semantics in passwords: The role of dates," in *Proceedings of the Ninth International Symposium on Visualization for Cyber Security*. ACM, 2012.

[22] L. Wilkinson and M. Friendly, "The history of the cluster heat map," *The American Statistician*, 2012.

[23] Z. Gemignani. (2010, April) Better know a visualization: Parallel coordinates. [Online]. Available: http://www.juiceanalytics.com/writing/writing/parallel-coordinates

[24] J. J. Miller and E. J. Wegman, "Construction of line densities for parallel coordinate plots," *Computing and graphics in statistics*, vol. 36, 1991.

[25] J. Nielsen, "Heuristic evaluation," *Usability inspection methods*, vol. 17, no. 1, 1994.

[26] J. K. Bradley, A. Kyrola, D. Bickson, and C. Guestrin, "Parallel coordinate descent for l1-regularized loss minimization," *arXiv preprint arXiv:1105.5379*, 2011.

[27] S. H. Hall. (2015, November) 7 case studies for understanding and using heatmaps. [Online]. Available: https://blog.crazyegg.com/2015/11/19/understanding-using-heatmaps-studies/

[28] P. G. Kelley, S. Komanduri, M. L. Mazurek, R. Shay, T. Vidas, L. Bauer, N. Christin, L. F. Cranor, and J. Lopez, "Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms," in *2012 IEEE Symposium on Security and Privacy*. IEEE, 2012.

[29] S. Komanduri, R. Shay, P. G. Kelley, M. L. Mazurek, L. Bauer, N. Christin, L. F. Cranor, and S. Egelman, "Of passwords and people: measuring the effect of password-composition policies,"

in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2011.

[30] R. Wash, "Folk models of home computer security," in *Proceedings of the Sixth Symposium on Usable Privacy and Security*. ACM, 2010.

Part I
Appendix

A ZIP Content

The ZIP file provided with this submission on SurreyLearn contains all the source files required to run this tool along with some of the password frequency and language lists. Note not all are included due these being freely available online and the fact they are large in size.

Prerequisites required to run the system locally To run the tool on your own machine it is required to have installed a HTTP server, such as XAMPP (https://www.apachefriends.org/index.html) and have the source code placed into the servers document root directory. Once the reader has a local server running follow these instructions:

Note: For the best user experience it is recommended that the reader use an up to date version of the Google Chrome browser.

1. Unzip cd.zip.
2. Place source/ into local server root directory.
3. Open a new Google Chrome tab
4. Navigate to localhost:PORT_NUMBER/source/index.html

Once step 4 has been completed the reader may use the functions described in the Implementation section by uploading password lists and dictionaries from cd/data/*.

Live running tool Along side this report a live and functioning domain hosts the most recent version of the tool, currently this is password protected for privacy reasons. The domain to the live tool is hosted on Google App Engine at http://paracoord-gapp.appspot.com/ and requires the following credentials to be used:

Username: assessment@surrey.ac.uk

Passcode: Paracoord#2016

Please note these credentials may be revoked and/or the domain name changed without notice. In this case please contact gary.read@gready.co.uk with the subject heading Dissertation Credentials to discuss further.

B Source Code

B.1 Default User Configuration JSON

The following code listing describes the default user configuration used by the tool, and are loaded and modified every new run of the program.

Listing 1: User Configuration

```
var config = {
    "debug" : true,
    "cw": 10, //cell width
    "ch": 10, //cell height
    "kh": 15, //legend height
    "hh": 15, //header height
    "hw": 15, //header width
    "sbh": 25, //Strength bar height
    "hish": 25, //Histogram height
    "fs": 12, //font size
    "flipy": false, //invert y axis
    "maxLength" : 15, //displayed as y-axis (-1 for all)
    "charSet" : "alphaNum", //display as y-axis
    "colourSet" : "YlGn",
    "impossibleCells" : "grey",
    "colourMode" : "inverseAlpha",
    "alpha" : 2.5,
    "axisOrder" : "alphabetical",
};
```

B.2 Default User Options JSON

The following code listing describes the default user options used by the tool, and are loaded and modified every new run of the program.

Listing 2: User Options

```
var options = {
    "tableSize" : {
        "min" : 5,
        "max" : 20,
    },
    "charSet" : {
        "lowAlpha" : "abcdefghijklmnopqrstuvwxyz",
        "highAlhpa" : "ABCDEFGHIJKLMNOPQRSTUVWXYZ",
        "alpha" : "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ",
        "num" : "0123456789",
        "lowAlphaNum" : "0123456789abcdefghijklmnopqrstuvwxyz",
        "symbols" : "! \"£$%^&*()_+-=[]{};'#:@~,.\\/<>?\|`¬",
        "alphaNum" : "0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ
            ",
```

```
14          "alphaNumSymbol" : "0123456789
                abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ! \"£$%^&*()_
                +-=[]{};'#:@~,.\\/<>?\|`¬",
15      },
16      "colourSet" : {
17          "Reds" :["#fee5d9","#fcae91","#fb6a4a","#de2d26","#a50f15"],
18          "Greys" :["#f7f7f7","#cccccc","#969696","#636363","#252525"],
19          "PuOr" :["#e66101","#fdb863","#f7f7f7","#b2abd2","#5e3c99"],
20          "BrBG" :["#a6611a","#dfc27d","#f5f5f5","#80cdc1","#018571"],
21          "PRGn" :["#7b3294","#c2a5cf","#f7f7f7","#a6dba0","#008837"],
22          "PiYG" :["#d01c8b","#f1b6da","#f7f7f7","#b8e186","#4dac26"],
23          "RdBu" :["#ca0020","#f4a582","#f7f7f7","#92c5de","#0571b0"],
24          "RdGy" :["#ca0020","#f4a582","#ffffff","#bababa","#404040"],
25          "RdYlBu" :["#d7191c","#fdae61","#ffffbf","#abd9e9","#2c7bb6"],
26          "Spectral" :["#d7191c","#fdae61","#ffffbf","#abdda4","#2b83ba"],
27          "RdYlGn" :["#d7191c","#fdae61","#ffffbf","#a6d96a","#1a9641"],
28          "Accent" :["#7fc97f","#beaed4","#fdc086","#ffff99","#386cb0"],
29          "Dark2" :["#1b9e77","#d95f02","#7570b3","#e7298a","#66a61e"],
30          "Paired" :["#a6cee3","#1f78b4","#b2df8a","#33a02c","#fb9a99"],
31          "Pastel1" :["#fbb4ae","#b3cde3","#ccebc5","#decbe4","#fed9a6"],
32          "Set1" :["#e41a1c","#377eb8","#4daf4a","#984ea3","#ff7f00"],
33      },
34      "impossibleCells" : {
35          "grey" : 'Grey',
36          "white" : 'White',
37      },
38      "colourMode" : {
39          "linear" : 'Linear',
40          "alpha" : 'x^alpha',
41          "inverseAlpha" : 'x^(1/alpha)',
42      },
43      "axisOrder" : {
44          "alphabetical" : "Alphabetical",
45          "frequency" : "Frequency",
46      },
47      "alpha" : {
48          "min" : 1,
49          "max" : 10,
50          "step" : 0.1,
51      },
52      "maxLength" : {
53          "min" : -1,
54          "max" : 60,
55          "step" : 1,
56      },
57      "flipy" : "false",
58  };
```

B.3 Corr2 Function

The following function is JavaScript implementation of the MatLabs Corr2 function. The function definition and algorithm used can be found here: http://uk.mathworks.com/help/images/ref/-corr2.html.

Listing 3: Corr2

```
var corr2 = function(A, B) {

      //Calculate minimal set of characters included in both A, B, and the shown
          charSet
      var minCS = [];
      var charSet = options.charSet[config.charSet].split('');
      var Ac = Object.keys(A.getState().charSet);
      var Bc = Object.keys(B.getState().charSet);
      charSet.forEach(function(c,i) {
        if (Ac.includes(c) && Bc.includes(c)) minCS.push(c);
      })

      //Calculate min maxLength
      var minML = Math.min(config.maxLength, A.getState().maxLength, B.getState().
          maxLength);

      //Calculate mean
      var A_freq = 0, B_freq = 0, maxFreq = 0;
      var B_mean = 0, A_mean = 0;
      for (var n = 0; n < minCS.length; n++) {
         for (var m = 1; m <= minML; m++) {
            //Frequency of character minCS[n] at position m
            A_freq = A.getState().position[m].chars[ minCS[n] ];
            B_freq = B.getState().position[m].chars[ minCS[n] ];

            //Max frequency used for normalisation
            maxFreq = Math.max(A_freq, B_freq, maxFreq);

            A_mean += A_freq;
            B_mean += B_freq;
         }
      }

      A_mean = A_mean / (minCS.length * minML);
      B_mean = B_mean / (minCS.length * minML);

      //Normalise mean
      A_mean = A_mean / maxFreq;
      B_mean = B_mean / maxFreq;

      //Calculating usign algorithm from http://uk.mathworks.com/help/images/ref/
```

```
            corr2.html
40      var A_nm = 0, B_nm = 0;
41      var A_mid = 0, B_mid = 0;
42      var AB_sum = 0, AA_sum = 0, BB_sum = 0;
43      for (var n = 0; n < minCS.length; n++) {
44          for (var m = 1; m <= minML; m++) {
45              A_nm = A.getState().position[m].chars[ minCS[n] ];
46              B_nm = B.getState().position[m].chars[ minCS[n] ];
47
48              A_mid = A_nm - A_mean;
49              B_mid = B_nm - B_mean;
50
51              AB_sum += A_mid * B_mid;
52              AA_sum += A_mid * A_mid;
53              BB_sum += B_mid * B_mid;
54          }
55      }
56      var r = AB_sum / Math.sqrt( AA_sum * BB_sum );
57
58      return r;
59  }
```

B.4 Handle DragEndEvent Function

The following function deals with the drag and drop behaviour function where an heat map is dropped onto another heat map.

Listing 4: handledragend

```
1  function handleDragEnd(e){
2      if (heatmap == draggedHeatmapUI) {
3          //Could do some fancy effect?
4      } else {
5          var scaleFactor = 1;
6          var wordCount = heatmap.getState().wordCount;
7          var words1 = heatmap.getWords();
8          var words2 = draggedHeatmapUI.getWords();
9
10         //Normalise word frequency from draggable if true
11         if (e.srcElement.id == "btnNormalMerge") {
12             var keys = Object.keys(words2);
13
14             for (var i = 0; i < keys.length; i++) {
15
16                 var word = keys[0];
17                 var oldfreq = words2[word];
18                 var newFreq = Math.round(getScaledValue(oldfreq, wordCount, 1));
19
20                 if (newFreq == 0) { //freq was the same
```

```
                words2[word] = oldfreq;
            } else {
                words2[word] = newFreq;
            }
        }
    } else if (e.srcElement.id == "?") {
        //Some new drag and drop feature button
    } else {
        return undefined;//Continue to add raw data
    }

    //Create new heatmap
    var newhm = new Heatmap("COMBI",undefined,undefined);
    newhm.buildHeatmap(words1);
    newhm.buildHeatmap(words2, true);

    //Create new heatmapUI
    var newhmui = new HeatmapUI(newhm,parentui,config,options);

    //Get parentui to draw this new combined hm
    parentui.uploadComplete(newhmui);
}

d3.selectAll('#dragoverUI').attr('class','hidden');

e.preventDefault();
}
```

B.5 Password Strength Function

The following function is used to calculate the strength of a password relative to the heat map its being calculated from.

Listing 5: getStrength

```
var STRENGTH_SCALE_FACTOR = 6;

var getStrength = function(word) {
    var strength = {};

    var avgFreq = 0;
    for (var x = 0; x < word.length; x++) {
        var pos = x + 1;
        var c = word[x];
        var freq = heatmap.getState().position[pos].chars[c];

        if (freq == undefined) {
            return undefined;
        } else {
```

```
15              avgFreq += freq;
16          }
17      }
18      avgFreq = avgFreq/word.length;
19
20
21      strength.avgFreq = avgFreq;
22      strength.value = getScaledValue(avgFreq, ui.svg.maxFreq, STRENGTH_SCALE_FACTOR);
23      strength.colour = getScaledColour(avgFreq, ui.svg.maxFreq);
24      return strength;
25  }
```

B.6 Heat map Class

The following class, heat map.js, has been included due to the highly functional aspects implemented within and for referencing within the document. The other classes are not as essential for the reader to review due to the live demo and screenshots, as they specifically deal with the user interface. Please note non-essential functions, such as getters and setters, are not included and this code listing is not completely truthful to the source code provided.

Listing 6: heatmap

```
1   var Heatmap = function(filename, state, parent) {
2
3       var defaultState = function () {
4           return {
5               name : filename,
6               wordCount : 0,
7               charCount : 0,
8               maxLength : 0,
9               maxCharFreq : 0,
10              uniqueCount : 0,
11              removedWords : {},
12              charSet : {},
13              position : {},
14              parent : undefined,
15              child : undefined,
16          }
17      };
18
19      var state = (state == undefined) ? defaultState() : state;
20      state.parent = (parent == undefined) ? undefined : parent;
21
22      /**
23       * { password : frequency, ... }
24       */
25      var buildHeatmap = function (rawData, append) {
26          //Append to HM or reset
```

```
27        if (append == false) state = defaultState();
28
29        //stats
30        for (pw in rawData) {
31            rawData[pw] = rawData[pw];
32
33            state.wordCount += rawData[pw];
34
35            state.uniqueCount++;
36
37            if (state.maxLength < pw.length) state.maxLength = pw.length;
38
39            state.charCount += pw.length * rawData[pw];
40
41            for(i = 0; i < pw.length; ++i) {
42                state.charSet[pw[i]] == undefined
43                    ? state.charSet[pw[i]] = rawData[pw]
44                    : state.charSet[pw[i]] += rawData[pw];
45            }
46        }
47
48        //Build this
49        for (i = 1; i <= state.maxLength; i++) {
50            if (state.position[i] == undefined) {
51                state.position[i] = {
52                        chars : {},
53                        words : {}
54                };
55            }
56
57            for (c in state.charSet) {
58                if (state.position[i].chars[c] == undefined) {
59                    state.position[i].chars[c] = 0;
60                }
61            }
62        }
63
64        //Populate this
65        for (pw in rawData) {
66            var len = pw.length;
67
68            for (cpos = 0; cpos < pw.length; cpos++) {
69                var pos = cpos + 1;
70                var c = pw[cpos];
71
72                //Character frequency
73                state.position[pos].chars[c] += rawData[pw];
74
```

```
75              //Max char freq
76              if (state.maxCharFreq < state.position[pos].chars[c]) {
77                  state.maxCharFreq = state.position[pos].chars[c];
78              }
79
80              //Add word
81              if (pos == len) {
82                  state.position[len].words[pw] = rawData[pw];
83              }
84          }
85      }
86
87      return this;
88  };
89
90  /**
91   *
92   */
93  var filterPin = function(n) {
94      var child = new Heatmap(state.name, null, this);
95
96      //user defined pin length or default 4
97      var n = (n == undefined) ? 4 : n;
98
99      var pttn = new RegExp("^[\\d]{" + n + "}$");
100     var words = getWordsOfLength(n);
101
102     for (var key in words) {
103         if (!pttn.test(key)) delete words[key];
104     }
105
106     child.buildHeatmap(words);
107
108     state.child = child;
109     return child;
110 }
111
112 /**
113  *
114  */
115 var filterDate = function(pttn) {
116     var child = new Heatmap(state.name, null, this);
117
118     var words = getWords();
119
120     if (pttn == undefined) {
121         for (var key in words) {
122             if (Date.parse(key) == null) delete words[key];
```

```
123             }
124         } else{
125             for (var key in words) {
126                 if (Date.parseExact(key, pttn) == null) delete words[key];
127             }
128         }
129 
130         child.buildHeatmap(words);
131 
132         state.child = child;
133         return child;
134     }
135 
136     /**
137      *
138      */
139     var filterThoseIncluding = function (arr) {
140         var child = new Heatmap(state.name, null, this);
141 
142         arr.forEach(function(w) {
143             child.buildHeatmap(getWordsIncluding(w));
144         });
145 
146         state.child = child;
147         return child;
148     }
149 
150     /**
151      * Remove a list of words from a clone of this heatmap, set as child.
152      * @return Return child heatmap
153      */
154     var removeWords = function(wArr, matchWord) {
155         var child = new Heatmap(state.name, null, this);
156         child.buildHeatmap(getWords());
157         state.child = child;
158 
159         out(child)
160         for (var i = 0; i < wArr.length; i++) {
161             var w = wArr[i];
162             child.removeWord(w);
163         }
164 
165         child.calculateStats();
166 
167         return child;
168     }
169 
170     /**
```

```
171     * Remove word from Heatmap, add it to removedWords for safe keeping.
172     * @return 1 for successful removal, 0 for not found
173     */
174    var removeWord = function(w, matchWord) {
175
176        //Check word exists in this heatmap
177        if (state.position[w.length] == undefined) {
178            return 0;
179        } else if (state.position[w.length].words[w] == undefined) {
180            return 0;
181        }
182
183        //Add removed word to removedWords
184        state.removedWords[w] = state.position[w.length].words[w];
185
186        //Delete word from heatmap
187        delete state.position[w.length].words[w];
188        return 1;
189    }
190
191    /**
192     * Get {word:frequency} array of all words using the following rules:
193     * Returns words of length n
194     * Returns words of length n to m
195     * @return Object dictionary of word objects
196     */
197    var getWordsOfLength = function(n, m) {
198        var words = {};
199
200        if (m == undefined) {
201            words = state.position[n].words;
202        } else {
203            //For each position
204            for (i = n; i <= m; i++) {
205                //For each word add it and its freq
206                for (w in state.position[i].words) {
207                    var wfreq = state.position[i].words[w];
208                    if (words[w] == undefined) {
209                        words[w] = wfreq;
210                    } else {
211                        words[w] += wfreq;
212                    }
213                }
214            }
215        }
216
217        return words;
218    }
```

```
219
220     /**
221      * Get {word:frequency} array of words including the string x
222      * @return list of objects {key:value}
223      */
224     var getWordsIncluding = function(x) {
225         var wordList = {};
226
227         //For each length
228         for (i = x.length; i <= state.maxLength; i++) {
229
230             //For every password of length i
231             var words = getWordsOfLength(i);
232             for (word in words) {
233
234                 //Add word if words includes x
235                 if (word.includes(x)) {
236                     var freq = words[word];
237                     wordList[word] = freq;
238                 }
239             }
240         }
241
242         return wordList;
243     }
244
245     /**
246      * Get this heatmaps character set ordered by their frequency
247      * @return Ordered character array by frequency
248      */
249     var getCharSetOrderByFrequency = function() {
250         var charSet = state.charSet;
251         var orderedKeys = []
252         var charSetArr = []
253
254         orderedKeys = Object.keys(state.charSet).sort(function(a, b) {
255             return state.charSet[a] - state.charSet[b]
256         });
257
258         for (var i = 0; i < orderedKeys.length; i++) {
259             var c = orderedKeys[i];
260             var cfreq = charSet[c];
261             charSetArr.push({[c] : cfreq});
262         }
263
264         return charSetArr.reverse();
265     }
266
```

```
267     /**
268      * Get this heatmaps word set ordered by its frequency
269      * @return Ordered character array by frequency
270      */
271     var getWordsOrderByFrequency = function() {
272         var words = getWords();
273         var orderedKeys = []
274         var wordsArr = []
275 
276         orderedKeys = Object.keys(words).sort(function(a, b) {
277             return words[a] - words[b];
278         });
279 
280         for (var i = 0; i < orderedKeys.length; i++) {
281             var w = orderedKeys[i];
282             var wfreq = words[w];
283             wordsArr.push({[w] : wfreq});
284         }
285 
286         return wordsArr.reverse();
287     }
288 
289     var getRemovedWords = function() {
290         return state.removedWords;
291     }
292 
293     /**
294      * Revealing objects and functions
295      */
296     return {
297         buildHeatmap : buildHeatmap,
298         calculateStats : calculateStats,
299 
300         filterPin : filterPin,
301         filterDate : filterDate,
302         filterThoseIncluding : filterThoseIncluding,
303 
304         removeWord : removeWord,
305         removeWords : removeWords,
306 
307         getState : getState,
308         getChild : getChild,
309         getParent : getParent,
310 
311         getWords : getWords,
312         getWordsOfLength : getWordsOfLength,
313         getWordsIncluding : getWordsIncluding,
314         getRemovedWords : getRemovedWords,
```

```
315
316         getCharSetOrderByAlphabet : getCharSetOrderByAlphabet,
317         getCharSetOrderByFrequency : getCharSetOrderByFrequency,
318         getWordsOrderByAlphabet : getWordsOrderByAlphabet,
319         getWordsOrderByFrequency : getWordsOrderByFrequency,
320     };
321 };
```