January Deliverable

Introduction

This report is for the design and implementation of an AI assisted web browser, it discusses the current progress and future work at the halfway stage of the process. The project firstly relied on a strong foundation in which essential functionality for any web browser must be in place before any advanced AI technology can be implemented.

The initial aims of the project were to design a web crawler that traverses the web in order to create a ‘crawl list’ of webpages that are relevant and up to date. The URL, title, summary, keywords and page rank of each website are extracted or calculated during this process and stored in a database so that they can be used at a later date. From here, a search engine will use the crawl list to return a list of relevant web pages based on a user query. These two sections are purposefully kept separate in terms of when they’re run so that the user doesn’t have to be concerned with or have any knowledge about the web crawler. This ensures that a large amount computation is done while the web crawler is running and not while the user is interacting with the system. However, it is not possible to perform all AI algorithms while the web crawler is running because specific user information is often needed and its important that the data on a user is changed during the interaction as well in order to keep it up to date.

Along with basic principles previously defined, a range of AI technologies will be implemented in order to enhance the user’s search experience. Firstly, a natural language processing (NLP) model will be used to read the content of webpages and create a summary and a list of keywords that will be used to help identify matches to a user query. This will be utilised by the web crawler which will help decrease the amount of processing that is done while the user is interacting with the system. The processing of user interactions will be a very important feature in creating additional functionality that will help the user in their searches. For example, by monitoring the websites that a user visits and recognising their activity on these websites, a user profile can be created. This will describe what the user’s browsing habits are and therefore can be used to provide customised suggestions, autofill search queries and promote new content that they might find useful.

In general, the purpose of this project is to create a web browser that gives the user an efficient, intuitive and easy to use system that will allow them to make the most out of their time spent searching the internet. This is made possible by the use of several different AI technologies that will collect and use information about the user in positive ways.

Progress Made

Web Crawler

As stated in the introduction section, the project relies on a strong foundation in which the basic, yet fundamental sections are set up and put in place before the more complex areas can be implemented. For this reason, the primary work was on creating an efficient and effective web crawler that could crawl the internet and create a list of relevant websites and a substantial amount of information about each one. This information includes the URL, title, keywords and a summary that can be stored in a website database and retrieved by the search engine whenever the user makes a query that is a match to that website.

The method that the web crawler takes to create and store the crawl list is as follows: An internet search is performed on each of the elements of some list called the 'queryList' which is specified before the execution of the code. In this searching process, a fake user agent is generated and sent with the headers of the HTTP request to mimic human behaviour and prevent the possibility that the program will be blocked by a website for appearing to be a bot. This is done by using an imported library called "fake_useragent" to create a random agent and adding it to the header of the request. However, this is not a fool proof method because some websites can use more sophisticated techniques to detect non-human activity on their sites, therefore it is important to have some error handling or fail safes to account for this possibility. Next, using the imported library "requests", an HTTP request is sent to search for the query with parameters for the URL, search parameters and the request header. A response is returned back, and after checking that the status code is 200 (if it does not then and error message is displayed), another library called "BeautifulSoup" is used to parse the html document that has been returned. In this process, all of the "<a>" tags (links to other webpages) are collected and verified to ensure that they start with "http". Once this has been determined, these links are sent to the web crawler. The web crawler is given specified values for the maximum depth and maximum number of pages that it can visit which ensures that a path of links has an endpoint and stops the crawling going forever. Next, the crawler visits each of the URLs in the list, extracting and returning the title, keywords and summary from each one. To produce the keywords and summary of each website, the module "WesbiteSummariser" is called.

The process begins with fetching the HTML content of the webpage using the requests library and parsing it with BeautifulSoup to extract text data from specific HTML tags such as paragraphs (p) and h1 to h6 tags, ensuring that it doesn't exceed a specified length. Subsequently, a summarization pipeline from the "transformers" library is employed to generate a concise summary of the text data which is given a maximum length set to 1024 characters. Simultaneously, the "spacy" library is used for natural language processing to identify and extract keywords. The number of occurrences of each word is counted, and keywords are selected based on a predefined frequency threshold. The code also uses a Merge Sort to sort the extracted keywords. The final result is a tuple containing the summary text and a list of sorted keywords.

Once all of this website data has been processed, an instance of the Website object is created so that the URL, title, keywords and summary can all be stored in one object and can have any operators applied to them as needed. To complete all attributes of a website, the relevance score and PageRank are calculated based on the extracted content and the number of meaningful links to and from each page. Only pages that have a relevance score of over 0.5 are added to the crawl list so that unimportant pages aren't added, keeping the crawl list small and more focussed.

Finally, all websites that have passed the tests have been accumulated into one list known as the “WebsiteList”, they are passed to a module called “SaveToCrawlList” that performs an SQL query to the website database to add in all websites to the database. This only happens as long as the entry doesn’t already exist. This marks the end of the current crawl cycle.

Web Browser

Front End

For the front end of the project, a search engine has been designed to allow the user to enter their search query into an entry box and a list of websites will be returned to them that match their search. Below each website title there is the link and a summary of the content of the webpage, this allows the user to get an overview of a website without visiting the site. The summary is generated and stored in the website database during the crawling phase to reduce computation time during the user experience.

Settings

The home page of the search engine also contains a settings menu where the user can sign in, create a new account, delete their account, view and manage their search history, and change their cookie preferences. Currently, cookie preferences are very limited and need more work because the use of cookies is very intricate and can affect many different areas of the program.

To create, sign in to or delete an account, a database called ‘users’ is queried to add a new entry, check an entry or remove an entry. Once a user is signed in, they will be able to change their preferences and view their search history from the settings menu because these are linked to a user account. A user can also delete any search history entries by simply pressing a delete button associated with the entry.

Additionally, once a user is signed in, their search history will be tracked while they are browsing websites which allows the search history database to be kept up to date. This is done by tracking mouse clicks, and the timestamp of these clicks, on any URLs shown to the user after they make a search.

Cookies

One of the most common features found throughout the code is the tracking and use of user cookies. When a device first accesses the website, a cookie consent prompt is shown, this provides detailed information about what cookies are used by the site and their uses, giving the user the chance to disable any of them if they want. Once this has been decided, a cookie called ‘visittedBefore’ is set to true and this consent form will not be shown until the cookie expires in a year’s time. Additionally, different cookies representing each consent given by the user are set to true or false depending on the answer given, these are used in a number of areas of the program to determine whether a certain piece of information is stored or not. Another cookie that is used is the username cookie which makes note of the current user that is signed into the system, and when an action like deleting the account or clicking on a new link so that the user can be found in the database and their information can be used correctly.

Search Results

After entering their search, the user is taken to ‘RetrieveWebsiteResults’, where PHP code first uses the search query and username (if possible) to retrieve a list of websites that match the query. This process involves comparing the search query to the title and keywords of all websites in the website list and determining if there is a strong enough match for the website to be ‘relevant’ to the search. This is done by setting a threshold value that defines the maximum differences that are allowed between to strings, if the differences are less than the threshold then the website is added to the list that is returned to the main page.

This list is then put into a table that displays the title as a link that can be clicked by the user to take them to the page, the URL, and a summary of the website that allows the user to understand what the website is about without clicking on it.

To reduce the amount of data that is sent back to the main page at any one time, pagination is used to split the website list into pages of a set size (currently this is set to 15). The websites are ordered by their relevance and the most relevant pages are sent over first while the others aren’t. If the user wants to look at the other pages, then they can do so by changing the current page at the bottom of the program. Using these techniques ensures that only a subset of the website list is transmitted from the server to the client at a time, and this is particularly important if the query has lots of matches and the website list is very large so that the processing and transmission time isn’t very long.

As previously mentioned, an onclick listener is used throughout this section to track all URL clicks performed by the user. These clicks are stored in the website history database with the timestamp and the current user.

Databases

Lots of information is needed at any one time in the running of the web crawler and browser. A database for crawled websites called ‘websites’ stores title, URL, summary, keywords and relevance score. The crawler writes to this database and the browser reads from it to return the list of relevant webpages to the user after a query. The ‘users’ database stores first name, last name, username, password and data of birth for any registered user of the system. The ‘search history’ database stores the URL, timestamp and username for each website visit that occurs while a registered user is signed into the system.

Draft of Contents Table

Abstract

1 Introduction

1.1 Goals

1.2 Why Undertake the Project

2 Literature Review

2.1 History of Web Browsers

2.2 Web Crawling

2.3 Search Engine

2.4 Mobile Web Browsers

2.5 Accommodating Designs

2.6 Cookies

2.7 Web Browser Security

2.8 AI Methodologies

2.9 AI Uses in Web Browsers

2.10 Development Process Models

2.10.1 Waterfall

2.10.2 Spiral

2.10.3 V Model

2.10.4 Incremental

2.10.5 Agile

3 Abbreviations and Acronyms

4 Additional Transferable Skills

5 Project Journey???

6 Design

6.1 Overview

6.2 Privacy Settings

6.3 Search History

6.4 Database Designs

6.4.1 Users

6.4.2 Websites

6.4.3 Search History

6.5 Classes

6.5.1 Website

7 Implementation

7.1 Web Crawler

7.2 Search Engine

7.2.1 Register

7.2.2 Sign In

7.2.3 Log Out

7.3 AI Data Extraction

7.3.1 Cookies

7.4 AI Data Processing

7.5 AI Data Uses

8 Results

9 Interpretation of Results

10 Figures and Tables

11 Conclusion 29

12 Future Work

13 References

This is not a finished version and there will undoubtably be changed throughout the next few months when new ideas are used or a change occurs in the approach to implementation or testing. However, it is based off of the current data that is forms the basis for the key areas of the project and therefore will help when working towards the completion of a specific section and the project as a whole.

Literature Review

Due to the importance of web crawling and search engines, there is a good amount of literature and research available that gives detailed explanations on one or multiple methods for each of these topics. Since the first PageRank algorithm was proposed by Larry Page and Sergey Brin, the founders of Google, in 1988 (Yu, Li and Zeng, 2021), computer scientists have worked on ways to improve and refine the method in which their algorithm goes about following the links contained in webpages and using these to determine the most relevant and useful webpages to a user. Hence, there are multitudes of these improvements documented but it is important to know that the context in which a specific algorithm is used can, and mostly likely will, determine how it works.

The purpose of this literature review is to explain the methodologies and techniques developed and used by software developers over many decades in order to reach the level of understanding and complexity that we see in the web browsers we use in our everyday lives. As well as this, the review will form the basis of explaining my reasoning and the logic behind the choices of implementation strategies used throughout my project.

2.1 History of Web Browsers

The Web was first invented in 1989 by Tim Berners-Lee. It was initially designed to provide for the demand for information-sharing among institutions and universities around the world and as a result this model only offered very basic functionality. Only reading data was possible because there was no way for a person to post themselves or interact with the content. The technologies used were HTML, HTTP, URI and JavaScript, along with protocols like XML, XHTML and CSS. Overall, the Web 1.0 was extremely slow and lacked a lot of functionality due to it only working in one direction.

In 1999, Darci DiNucci released the Web 2.0 (also called The Participative and Social Web) but it didn't become widely used until 2004 and it is still the current version of the World Wide Web used today. It allowed a user to post content and, as an extension, connect with other people also using this new Web (meaning it was bi-directional, a significant difference compared to the Web 1.0) which was revolutionary and created a wide range of new possibilities. The introduction of technologies such as XML, DOM and REST allows a creator to include interactivity and communications with a client. However, with the introduction of these new functionalities, insecurity also crept into the infrastructure of the Web, specifically, they allowed a user to be hacked because a person can add their own, potentially malicious, programs that can be a threat to another user. Therefore it has become essential for developers to think carefully about how they can ensure user safety when interacting with their website. (Shakir M. Abass, 2019)

Web 3.0 is a theoretical model that researchers believe is the next iteration of the Web. It is a set of values and applications that define the new age of the World Wide Web, but it is only a prediction on what people expect the current version to evolve into in the future and therefore it is hard to say for sure what the final product will look like. There is evidence of Web3 in use today, for example the Metaverse is an example of an immersive web that allows a user to interact even closer to the information that they want to find as well as other people on the Internet. Firefox now supports WebVR and A-Frame, which let developers build VR websites in faster times than ever, and this will only speed up the process of converting from a Hyper Text based web to an immersive one. The blockchain is a decentralized, distributed ledger that is used to record transactions with the help of a

large collection of computers ensuring that the transaction can't be altered or incorrectly recorded because all computers keep and record and can say when there has been an error. It is said that Web3 will be built on top of blockchain technologies so that this same principle of decentralization and error free transactions can be applied to create a faster, more secure platform. (*What Is Web 3.0 (Web3 definition)?*, no date)

It is no surprise that the first graphical web browser came at the same time as Tim Berners-Lee released Web 1.0 to the public because a person using the web would need a way to search through all the data stored to find what they were looking for. He called this access point to the internet the Worldwide Web and a year later, a Line Mode browser was created to allow internet access through basic computer terminals.

As the popularity of this browser grew, companies saw a big opportunity and began thinking of ways in which they could benefit from it. In 1993, Mosaic was created at the NCSA, which came exceedingly popular and is the early ancestor of Mozilla Firefox. Then, in 1994, Andreessen released Netscape Navigator, which was the first step towards a so called 'Browser War'. Following this in 1995, Windows licensed the old Mosaic code and released its own Internet Explorer and from then on Netscape and Microsoft competed with each other to create the best browser.

A big milestone that came out of this war was JavaScript, invented and released by Netscape, which gave websites lots more computing capabilities and functionalities. Shortly after, Microsoft was responsible for the release of CSS that helped style webpages to make them easier to use and gave them a nicer overall appearance. But when Microsoft started to equip all their Windows operating systems with Internet Explorer, their dominance was quickly established and by 1999 they controlled 99% of the market. In 2004 Netscape open sourced its codebase to create Mozilla, a non-profit web browser that gave users a choice. By 2010, Internet Explorer's market share had dropped down to 50%. Other competitors arose around this time as well, such as Safari and Google Chrome, further reducing the popularity of Internet Explorer. In 2015, Windows 10 saw the replacement of Internet Explorer with Microsoft Edge.

(*Browser History: Epic power struggles that brought us modern browsers*, no date)

2.2 Web Crawling

Web Crawling has been widely used for decades now due to the need to extract copious amounts of data from the Internet. Due to the nature of the format of data, namely that it is unstructured, it can be a challenge to take away the valuable information that is needed for a specific purpose. Web scraping has many valuable uses in a wide range of areas such as: a price comparison site looking at different company's offers for a quote, analysis of social media data, or making accurate stock market predictions. But in the context of a web browser, a web crawler simply has to follow a series of websites by looking at the hyperlinks that are present in each one so that it can record websites that it finds on its way.

The challenge is creating a foolproof method that allows the crawler to handle websites where they will have to enter a username and password. It is not possible to code a crawler that has a valid account for any site that it may come across and it is equally as impossible for the crawler to have the ability to guess the correct details to get passed this security. And so a crawler is usually given input directives that allow it to “assign specific values to specific input fields” (Liu, Chen and Sun, 2020) so that the crawler can apply these input directives in fields as needed. A number of issues arise through the use of input directives. Namely, it is very difficult for a developer to know what the format of the fields will be like in an unknown website and so it is incredibly difficult to design the crawler in a way that will be able to deal with an unknown format, and it will be necessary to analyse the text in a webpage in order to figure out the required inputs which is a tedious and time consuming process.

2.3 Search Engine

For the most part, all examples of modern search engines that you can find on the internet (e.g. Google, Bing, etc) all follow the same format and principles. There will always be the name of the search engines and the search bar as the main two features of the page. These are the essentials for a webpage GIVE AN EXAMPLE OF A SOURCE THAT SUPPORTS THIS. Additionally, it is common to see the websites that the user most frequently visits so that they can have quick and easy access to then without the need to type them in the search bar. Following on from this concept of remembering search history, when a user starts to type a query into the search bar, lots of existing web browsers try to predict what they want to search and will autofill or give personalised suggestions to try and speed up the search process.

It is also common practice for a web browser to offer a settings page where the user can set any preferences that they would like. For example, switching between light and dark mode (but this may also be offered by the application or operating system that they are using as well), or whether the user is ok with the storing of their relevant search data. Additionally, a user may want to delete their search history, they may want to remove all of it or just a specific page or time interval and so it is very common to see an option to facilitate this in the settings page of a web browser. A more advanced feature that can be offered to the user is the ability to turn on safe search where the user is protected from any malicious data or programs on websites that can cause harm to their device. Protection like this can be achieved by using a blacklist to mark pages as dangerous, or by scanning the content of a website to find any potential threats that the user should be warned about and blocked from accessing (Hr *et al.*, 2020). An obvious extension of this is to create a child-friendly search environment in which a parent or guardian can be sure that their child won’t be able to access sites with dangerous or inappropriate content. Another choice that can be offered by the browser is the ability to enable or disable JavaScript in order to offer protection against security threats and malicious attacks. However, doing this comes at the expense of the user experience and so should be offered to the user only in situations where the page won’t greatly suffer. For example, if a page relied on JavaScript for animations and other interactive features then disabling these will reduce the helpfulness and readability (Das, 2022). Sometimes a website will want to use a user’s location to help with the service they provide, for example in a weather forecast or finding the closest shop to the device’s current location). This can obviously be very helpful, but some people might not feel comfortable with being tracked and having their location stored, therefore it is good to have an added feature that lets the user block any requests or attempt to locate their whereabouts. The same is true for other requests like using the device’s camera that a user might feel uncomfortable about.

Another very common practice in a search engine is the use of pagination. This is the process of splitting up the list of websites that have been shown to be some sort of match to the user's query into smaller groups and only displaying one group at a time. These groups are also called pages. On the webpage, the user is shown a sequence of numbers from one to the number of pages and has the option to click on any of these numbers to bring up the websites belonging to that page. This allows the user to see only a small number of webpages at a time so that they don't have too much data to look through at one time, whilst also providing insight into the relevance of a website by the page that it appears in. The best example of this feature in a search engine is Google's implementation of pagination. If you were to go to the bottom of the search results page after entering a query, you will see Google written with lots of o's and these o's are links to a specific page returned by the search engine.

2.4 Mobile Web Browsers

Making any webpage compatible with a mobile device is a necessity for any web developer these days, and a search engine is no exception. Due to the rise of mobile devices, starting from the release of the first iPhone in 2007, mobile web browsing technologies have boomed in a very short span of time, and this has led to a number of competing versions to emerge. Among these are Google Chrome and Safari. "Many information technology companies have recognized the usage share of their mobile Web browsers as an important marketing strategy" (Ahn, Kim and Proctor, 2018) and therefore have made a significant effort to create the best UI for their browser. The many task that these companies have had to deal with is working with the greatly reduced screen size that comes with using a mobile device compared to a larger one (like a desktop computer), and so it is important to try and maximise usability, learnability and usefulness. (Ahn, Kim and Proctor, 2018)

It should be noted that although this is an important aspect of a web browser due to the magnitude of mobile devices in the world today, the main focus of this project is to implement additional machine learning and AI elements to the program and so I will not focus on it too much when it comes to the implementation stage.

2.5 Accommodating Designs

Another highly relevant area for website design in the modern world is incorporating an accommodating design to allow for easy access to any person with visual, audio or any other kind of impairment. Even though it does again fall outside of the scope of my project, it is good to acknowledge the extra design considerations that would go into a web browser that would be released for public use.

Increasing web accessibility for older adults (Sa-nga-ngam and Kiattisin, 2020). Study aimed to create a redesigned prototype web browser that allowed for, and can be proved to be, more user friendly for the older demographic of people (people over 65 years old which is around 10% of the world's population). As of 2018, 90% of the world used the Internet but on 78% of people aged 65 and over were reported to actively use the Internet. It is this divide in generations that has come to be a concern and a problem that Prush Sa-nga-ngam and Supaporn Kiattisin sought to analyse and find a solution for in order to provide support to older adults living in this new digital age. A few of the main findings of the study show that creating a more simplistic UI, adding assistive features (stopping ads, removing animations, hiding pictures, etc) and creating a new layout design were the most effective ways in which to improve user experience.

HCI and culture (Kyriakoullis and Zaphiris, 2016). A big challenge in the design of a user interface is to be accommodating to the diverse cultures that will potentially interact with the system. It is essential to understand cultural values to create a successful and widely accepted system, but this can often be an almost impossible task as there is so much that a designer and developer will have to consider. In fact, "research community examines those cultural characteristics that may partially be responsible for the slow uptake or rejection of a system" (Kyriakoullis and Zaphiris, 2016). The study found that the most important consideration HCI is the user interface because it is the first impression that the user gets of the website or system and so it can be as simple as the user rejecting it in the first few seconds of use. Marcus and Gould, for instance, report that it is very important to have a number of options to choose from in a high uncertainty avoidance culture in order to prevent users from getting lost in the system (Marcus and Gould, 2000). Additionally, it is particularly important to think about "invisible cultural attributes", specifically the cultural attributes that affect a person's or a society's behaviour. In a study by Su and Adams the difference between two e-commerce sites, Amazon tailoring to an international market and Dangdang tailoring specifically to the Chinese population, and how it is very hard to apply the Amazon business model to a Chinese market (Su and Adams, 2005). For example, it has been found that while the typical British culture is to consult family and friends before making a purchase, while the complete opposite is true in Chinese culture and an individual would rather go through with a purchase by themselves. This, among many other characteristics, show that there is a completely different view online shopping depending on the culture and therefore extra attention is needed when trying to expand or accommodate to a different culture.

People with cognitive difficulties will typically face a number of challenges when it comes to interacting with and using a piece of software and it can help to analyse the reasons for these challenges in order to design a system that allows a person to get the most out of it(Gregor and Dickinson, 2007). As opposed to designing a system with people with physical impairments in mind where we are helping with dexterity problems, designing the system for people with cognitive impairments requires the focus to be on how we can help those who most likely don't have the technical knowledge or skills to use the assistive features provided with many websites today. In a study, Dickinson et al. examined the useability of Microsoft Outlook Express and found that 50% of a group of older users failed to complete a list of basic email tasks, and it is this complexity that needs to be reduced in order to make a system more useable, especially when considering those with cognitive impairments.

2.6 Cookies

The GDPR says that the use of cookies to gather and store data to potentially create an identifiable profile of a user means that they qualify as personal data and therefore they are subject to the GDPR. As a result, this means that cookies should be stored as securely as any other piece of data stored by a company and said company is responsible for any breaches of this data (*Cookies, the GDPR, and the ePrivacy Directive*, 2019). Further legislation on cookies passed in 2002 and amended in 2009 in the ePrivacy Directive. Its most notable consequence was the proliferation of cookie consent popups. Its main purpose is to enhance the GDPR, which only contains one reference to cookies in Recital 30 (*Recital 30 - Online identifiers for profiling and identification*, 2018), in order to expand on the confidentiality of electronic communications and tracking internet users.

When cookies were first used on the Internet in 1994, they were designed to support added functionalities and improve the user experience on a webpage by remembering small details about interactions had by the user. But as time went on, companies realised the potential that cookies offered in terms of the collection and manipulation of user data and companies unlawfully processing this data has become quite a serious fear for many people. In the study by Alharbi et al., out of the chosen 200 websites, only 30% provided information to the user about what a third party does with the cookies that it extracts (Alharbi *et al.*, 2023). This is where the concern comes from for most people because there is not enough transparency and openness to the consumer about what data is being collected, how it is being user and the potential risks that are associated with the processing of said data. (Kumar and Sharma, 2015). There are regulations in place that are designed to enforce any provider or developer to be honest and abide by the 'privacy by design' principle rather than making the user do the work to keep their data safe. And so, a developer must now get the explicit request of the user in order to use coolies for the storing and processing of any data whatsoever.

The most acceptable practice when it comes to getting permission to the use of cookies is by allowing the user to accept or decline the collection of certain or all cookies. Sometimes cookies are labelled as 'essential' which means that they are needed for the general performance of the page and the website won't be useable without them, but this does not remove the need for a developer to get consent from the user before tracking them (*How do we comply with the cookie rules?*, 2023). Others are 'unessential', which means that they are used to store additional information about the user that are used to create a user profile for first or third parties. Unessential cookies should be able to be disabled by the user at any point in time if they decide that they don't want their data to be collected and this has to be complied with due to the ePrivacy Law. Third parties will use cookies for advertising to give the user relevant and targeted ads to pique the interest of a person into viewing a product or site. Doing so is well within the law, but a company must be careful with the ways in which the acquire the cookies. This means that they must be completely transparent with all of the ways that the cookies will be used and who will have access to them, and there should be no insecure handling of any user data. One famous example of where a company did not comply with cookie regulations was Google in 2021, who was heavily fined by the French data protection authority Commission, Nationale de l'Informatique et des Liberté **(CNIL), for the illegal use of cookies. On the French version of Google, the platform YouTube asked users to consent to the use of cookies in a way that was a lot easier to accept cookies than to reject them, this goes against the principle of 'privacy by design' and so Google was found guilty of the charges. There was a comparable situation with Facebook at the same time in which they were fined** €60 million and given three months to comply to the charges and change their system so that they fit legislation.

2.7 Web Browser Security

Blacklists (Virvilis *et al.*, 2015). Blacklists are a common tool that developers will use to mark a website as 'rogue' or untrustworthy. The purpose is to alert a web browser of a website that will likely cause harm to a user if they were to access it and therefore stop anyone from falling victim to the malicious program. There are many URL blacklists that can be used by developers, the most common being Google's Safe Browsing and Microsoft's SmartScreen, and these usually offer enough protection against common and traceable threats. The study from Virvilis et al. investigates the potential problems and shortcomings of the use of URL blacklists, and specifically finds that the effectiveness in which many anti-phishing tools find and block malicious sites was not up to the standard that it should be.

Phishing is a social engineering technique in which a hacker will pretend to be someone they're not or create a fake scenario in order to steal important information from a victim (Mazher, Ashraf and Altaf, 2013). As previously mentioned, using a blacklist to alert a browser of any malicious websites isn't enough to create a completely safe search environment and so extra protection is needed. This is where the heuristic-based approach can be put to use. In this technique, a programmer can try to determine if a site is malicious by checking for one of many suspicious signs. This is done by checking the URL authenticity and scanning the content of the webpage. Machine learning algorithms have been introduced to perform these tasks which has led to the speed at which a malicious site is found to increase.

The most common and well-known technique in improving the security of a web browser is the use of security toolbars. The provider of the search engine will often provide lots of different features that can help a user navigate through the internet and avoid any potentially harmful data or content. For example, if a user tried to download something that has been flagged or appears suspicious to algorithm, then an error message will popup asking the user if they are confident that what they are about to download is safe, or the browser may block the process altogether until the administrator of the system changes some permissions. A study performed by Mazher, Ashraf and Altaf investigated the effectiveness of how well a number of web browser performed in alerting a user of a potential threat on a website they are about to access. The browsers involved in the experiment were Internet Explorer, Firefox and Google Chrome. The results showed that out of the three browsers, only Google Chrome was successful in alerting the user to the threat that was posed by the phishing website they were trying to access. Even though this test was performed in 2013, the main principle of the test remains relevant to this day because it will always be important for a successful and safe web browser to detect threats to a user and alert them of it before it's too late. Furthermore, the results showed that for Internet Explorer and Firefox, all of the users proceeded to use the phishing website because they believed it to be the real version of Facebook, and because neither browser alerted them to any kind of threat, they entered all of their account information, opening up the possibility of them being hacked. This further emphasizes the importance of a web browser providing state of the art protection for any user.

A Quick Response (QR) code is used to represent data, like the address or URL for a website in a scannable form ready to be parsed by a consumer's mobile device. This technology has allowed providers to make their platform more easily available by putting a code in a heavily visited space where it can draw attention. However, hackers have seen QR codes as an opportunity to launch a new kind of attack on a victim's device. Commonly, if a consumer is told to scan a QR code in a leaflet for some produce then they will, even just out of curiosity, which gives hackers the opportunity to place links to malicious sites or data that will exploit the device of the scanner when accessed. Similarly, a hacker can also replace an existing QR code (for example, a code that has been pinned to a noticeboard) with their own, tricking any person who sees the QR code into thinking that it is a legitimate link when it is not. This type of cyber-attack is called QRishing (Vidas *et al.*, 2013). The study by Vidas et al. confirmed that the majority if people will scan a QR code out of curiosity rather than because they were interested in the relate information that the link contained, with 64%

scanning for this reason. An additional 14% scanned the code for fun, highlighting the fact that the majority of people don't consider the threat of hacking when scanning a QR code.

Another important consideration that a developer needs to make when it comes to website design in the secure transfer of data from the client to the server in order to store, process or retrieve information. There is the simple use of POST and GET in an HTML form that allows the programmer to specify the method in which the data is sent over to the server or another webpage and can be useful in the simple cases. The GET method sends the user information appended to the page request separated by a '?', whereas the POST method transfers information via HTTP headers so that the information is encoded and put into a header called 'QUERY_STRING'. Hence, the POST method is more secure than the GET method and is the only option for sending sensitive user information from an HTML form to a server to be processed.

However, there are a number of mechanisms and software that can be used to further increase the security of internet transactions. A key example for the use of a web browser is PDO. This is an extension that makes data stored in a database safer from any SQL injection attack made by a hacker trying to gain access to the system or data inside it by entering SQL commands into the input boxes of a website. This is a very simple yet effective approach to trying to access data without the permission of the provider. PDO can be an added barrier to protect against such attacks because a programmer can make use of prepared statements in their code for storing or manipulating data in a database so that it is harder for a hacker to use SQL queries as a malicious input.

TALK ABOUT OTHER SAFETY FEATURES I COULD USE IN MY CODE – DON'T HAVE TO USE THEM BUT GOOD TO TALK ABOUT THEM AND THEIR POTENTIAL BENEFITS

For the most part, creating an entirely secure search platform isn't the main focus of my project. I will of course add in basic functionalities to keep any data that I extract safe and out of the hands of any unauthorized persons but the overall security of the data that is stored will be down to the effectiveness of the server. Additionally, the program will most likely only be used on a small scale and so there isn't the concern that the data will get into the wrong hands under any circumstances. For this reason, with the exception of PDO (AND MAYBE OTHER EXTENSIONS I FIND AND USE), the information that I talk about in this section is outside the scope of the project.

2.8 AI Methodologies

A neural network is one of the most common ways in which large amounts of data is processed in order to be categorized. The definition of a neural network is 'a computer system modelled on the human brain and nervous system' (*Oxford Languages and Google - English | Oxford Languages*, no date), meaning that there are a large number of nodes (neurons) that are interconnected with one another to pass on data. The data is processed in such a way that a value is returned at the end that can be used as a prediction or classification for the input values that were given (Abiodun *et al.*, 2019). The first mathematical model, called the MP model, was devised by McCulloch and Pitts 1943 but never taken seriously because of its limited capabilities. Rosenblatt proposed the single layer perceptron which was based on the MP model but added learning capabilities (much like the human brain), but this new model still couldn't handle linear inseparable problems. Rumelhart et al. propositioned a multilayer feedforward network that is trained by the error backpropagation algorithm, and this formed the basis for all neural networks that we use today. (Li *et al.*, 2022)

A convolution neural network is a specialised form of neural network that operates most effectively on images. This has made it revolutionary for computer vision technologies such as face recognition, autonomous vehicles and intelligent medical treatments. However, this area of the topic is out of the scope of the project because there is no form of image recognition.

However, it is important to understand that neural networks come with potential drawbacks. For instance, a significant amount of pre-processing is required in order to make a data set useable by the model. This process involves data cleansing to fill in any missing or erroneous values, removing outliers, and data normalisation. The data must then be split into a training, validation and test set, and the way in which the data is split is very important because it could lead to skewed data in one or more of the sets causing one of the phases to fail. Any or all of these processes can be automated by some algorithm, and this is especially common when vast dataset are being used because it would be too time consuming for a person, or group of people, to try and sort through all of the data manually. However, this places a lot of trust on the algorithm which is not always sensible because any such algorithm is never guaranteed to perform perfectly without causing errors, and so a human would have to check through all of the data after it has been processed anyway to check for any errors (Mishra *et al.*, 2020).

After the pre-processing phase, a suitable model has to be developed in a way that produces the best output with the majority of data that it is given. This process is very computationally expensive and time consuming because it requires a trial and error approach to be taken in order to find the best configuration of nodes, edges and weights (Livingstone, Manallack and Tetko, 1997).

Due to the complexity of a neural network, it can often be almost impossible to analyse the way in which the output is calculated. The lack of knowledge for the causal relationships between nodes in the network can lead to misleading interpretations and predictions, especially in large and complex networks. The number of edges is dependent on the architecture of the network, but also dependent on the number of nodes and hidden layers used for some computation in the network. Let N=the number of nodes in the input layer, M=number of nodes in the output layer and L=the number of hidden layers (each with H[i] nodes) then the total number of edges in a typical neural network is

$E = N \times H[1] + (\sum (i = 1), (L - 1)\ H[i] \times H[i + 1]) + H[L] \times M.$

As shown, as the number of nodes or layers increases, the number of edges increases at a faster rate. This is a problem with very large and complex neural networks because the computation time increase exponentially, meaning that a large number of resources are needed for a project involving a large neural network (Garg, Jegelka and Jaakkola, 2020).

Furthermore, it is common for a neural network to lack in robustness, meaning that the network may not adjust well to different inputs of data. For example, if the model is trained on a certain dataset then it will only be familiar with the data in the set and other data points of a similar nature, so if a new data point is given to the model that doesn't fit within the normal ranges of the training data, then the network may incorrectly process or classify it. Similarly, with the user of image processing in a neural network, a small change to an image, like a change in the lighting, wouldn't change the perception of a human, but the network would be unsuccessful at processing the image if it hadn't been tested on an image with this level of lighting before (*What are the limits of deep learning?*, no date).

Another area of deep learning that is commonly found is Bio-Inspired AI. This field studies the behaviour of individuals and groups of animals found in different environments in the world to find new ways to go about solving problems. The Particle Swarm Optimisation method draws inspiration from the behaviour of a flock of birds trying to hunt a school of fish. As opposed to one bird trying to catch prey, a flock has the ability to search through a large search space and alert the individuals around them of any successes that they have during a certain time. This process can be replicated as a method used by an AI algorithm to find the optimal value in a search space. A 'swarm' is initialised, and each particle is allocated a random position and random velocity in the search space. At each increment, the position is updated based on the velocity of the particle, and a utility function is applied to evaluate how good the current position is in relation to some target utility. Each particle then looks at the utilities of its neighbours and updates its velocity in relation to closer areas of the search space that seem to give a high utility. After a large number of iterations, the particles will find an optimal point in the search space, this is known when the particle no longer has a velocity. This idea of group behaviour has many more applications and shows how emergent properties in a group are demonstrated by each individual learning and helping other individuals around them.

2.9 AI Uses in Web Browsers

The use of AI libraries and APIs plays a key role in the implementation of any machine learning related program due to the ease and vast capabilities that they provide. With a search engine, automation can be introduced into the system in a number of different areas in order to improve the user experience, increase security and maximise efficiency. For example, the use of machine learning algorithms to scan the content of webpages for threats (already discussed).

As a way to add convenience to the searching process for the user, a natural language processing (NLP) model can be used to scan the content of a website to create a summary that can be shown to the user. This way, a user can know what a webpage is about without having to spend time reading through it and therefore can save time and effort. One of the main NLP models is called BeautifulSoup which creates a parse tree for parsed webpages, giving the developer the ability to extract data from the HTML document. This makes it very useful for web scraping, allowing the developer to use elements in an HTML document as pieces of data to be processed. Other NLP models include LXml and RegEx with RegEx being the fasted but least scalable due to having limited extraction rules (Thivaharan., Srivatsun. and Sarathambekai., 2020).

Applications of neural networks in web browsers are moderately common, with the vast amount of data stored on a person on the Internet, it can be helpful to create ways to process the data in ways that benefit both the user and content providers. A single user will mostly likely have countless accounts on network services on the Internet and so linking these accounts into one, central area can be convenient and efficient for a user. This process is called identity linking (Qiao *et al.*, 2020). Identity linking has a number of other useful benefits as well such as grouping users into group with something in common. This could be identifying that a user is a minor and so should be restricted from certain content, or identifying a malicious user that intends to do harm to another user, so it is in the interest of the browser provider to stop this to prevent a cyber-attack on one of their users. In the study by Qiao et al. a Siamese neural network was used compare high level feature representation of the behaviour of web users. MAYBE ADD MORE – LINKS VERY CLOSELY TO MY PROJECT

A study by Kolakowska et al. sought to determine whether it is possible to track human computer interactions like keystroke dynamics and mouse movements to recognise the gender of a web browser user. Other studies have been made on the use of these human computer interactions, for example, recognising a user from their mouse movements (Pusara and Brodley, 2004), or detecting cognitive and physical stress through typing behaviour (Vizer, 2009), but none have gone as far as to accurately predict a key personal characteristic like a person's gender. The experiment involved recording the interactions with a mouse, keyboard, track pad and touch screen but only enough meaningful data was collected for mouse movements after the pre-processing phase. Results showed that the user of a rotation forest to process and categorise new data points yielded an 80.8% success rate, and a Bayesian Network gave a 79.12% success rate.

OTHER AI TECHNOLOGIES FOUND AND USED LATER ON.

2.10 Development Process Models

2.10.1 Waterfall

The waterfall development lifecycle is one of the simplest models created. The different steps are definition, design, implementation and unit testing, integration and unit testing, and deployment. Each step in the cycle must reach completion before the next to start, and once they have been marked as finished, it is not possible to return to that stage. This makes the waterfall mode static which means that it can't adjust to changing requirements. Its main uses are in very large projects because it can be useful to have a set list of requirements that the development team can work towards, and they can be confident that what they are making is exactly what has been previously described to them. However, other models can be a better fit for smaller teams. (Klopper, Gruner and Kourie, 2007), (Lock, no date)

2.10.2 Spiral

The spiral model develops software in a flexible and step-by-step manner, by repeating cycles, or "spirals," to gradually build and improve the software. Each cycle involves four key steps: setting goals and identifying risks, finding solutions and planning the next steps, building the software, and testing and evaluating the results. After each cycle, the new findings and learnings are used to update and improve the software. This process continues until the software meets all the requirements. The spiral model is like climbing a staircase, where each step brings the software closer to being finished, and any issues are addressed along the way. (Klopper, Gruner and Kourie, 2007), (Lock, no date)

2.10.3 V Model

This model uses similar ideas to the Waterfall except it highlights the relationship between each phase of development and its corresponding phase of testing. The left side of the "V" represents the development phases, and the right side represents the testing phases. The advantage of the V-Model is that it provides a structured approach to testing, ensuring that testing activities are planned and conducted at each stage of development. It also emphasizes early testing to catch and fix defects as early as possible in the development process. (Klopper, Gruner and Kourie, 2007), (Lock, no date)

2.10.4 Incremental

The system is divided into small manageable parts where each part works towards the overall functionality of the system. Development is performed incrementally, and at each increment new features are added, or existing parts are improved. This approach allows the gradual improvement of the system to be shown to shareholders and gives the opportunity for regular feedback that can be added into the next increment, which is especially good when the requirements aren't clear or initially misunderstood. However, this comes with increased costs and will be harder to manage. Additionally, it is more time consuming because in depth documentation is needed for each increment. (Klopper, Gruner and Kourie, 2007), (Lock, no date)

2.10.5 Agile

Agile development aims to make software in a flexible and collaborative way. Instead of following a rigid plan, it focuses on working closely with people and being open to change. The main idea is to deliver small and functionally useful parts of the software in regular intervals which is usually every few weeks. This allows improvements to be made quickly based on the feedback from users. Teams include different types of experts that communicate frequently and use methods like Scrum or Kanban to organize their work. They have regular meetings to discuss progress and solve problems together. The goal is to create a product that fits the needs of users better and is more adaptable to changes. (Leau *et al.*, no date), (Lock, no date)

Remainder of Work Required

In the project's current state, a user is able to effectively search through the crawl list by entering a query into the search engine, and a reasonably relevant list of websites will be returned to them giving them the ability to visit these websites if they choose to do so. However, the methods in which these websites are deemed to be relevant are quite simple and so in the next stage of development it is essential that a more complex set of algorithms is developed so that the list of websites returned to the user after a search is a relevant and useful as possible.

Another area in which improvement is needed is the tracking of URL clicks to record search history. Currently, the program can only record website visits that come directly from the search results page, but if a user is to click on a link after this point, then this will not be recorded. Therefore, the challenge is to find a way in which the mouse clicks can be tracked even when a user isn't on the search engine website anymore. A possible solution would be to load the content from a website into the search engine rather than taking the user to the new site, but this could come with difficulties and will need to be researched more.

One of the main components is the use of AI libraries and technologies to analyse user data to create user profiles that will be used to provide a more personalised experience and overall enhance the searching process. Features that have been researched and will be added in include predicting user queries, content recommendations (for example, showing webpages that have been deemed to be a better fit for the user than others), and improving security by detecting threats. Another possible feature that could be added is a personal chatbot used to give a user information to questions in very little time. This will eliminate the need for the user to visit websites because they can be told an exact answer to a question by the chatbot. However, this will be a very complicated process if it is to be implemented using completely original code and so it will be less problematic to use predefined chatbot libraries like the ChatGPT API.

Furthermore, the way in which user data is collected and stored will need to be improved in order to better fit the requirements of the program in a number of different areas. Currently, only information about the mouse clicks that the user makes when viewing search results is captured and this isn't enough data to satisfy the desired capabilities. Search history can be used to create a profile of the types of websites that the user frequently visits by determining the genre of each visit. With this information, personalised suggestions can be given. To further build on this notion, sub genres can also be analysed so that a more detailed and specific profile can be created for the user's interests and search habits. For example, if a user has recently been looking at healthy cooking recipes, then instead of just noticing that the user has been wanting to learn to cook, the program realises that the user wants the cooking to be healthy and can help point them in the right direction.

As previously stated, the storing and processing of cookies will be very important for the functionality of the website. More cookies will be introduced that will be used by the AI technologies to aid in giving the user a better experience. By tracking the amount of time that a user spends on a website, or creating ways to analyse how engaged a user is with some content, the AI will be able to determine if a user is interested in a certain topic or if maybe it is a task the user doesn't want to do but has no choice in the matter.

There is no plan to use any third party or advertisement cookies because the purpose of the project is not to create the browser for commercial use. However, if at any time there was a reason to add them to the code, then this could be done quite easily by relating them to the recent activity data.

Overall, now that the basic features required to make any web browser effective and useful for a user are completed, the main focus for the next stage of the project will be to further research AI technologies that can have a positive impact on the program and implement them accordingly. Once this has been done to an impressive level, tests will need to be conducted where real user tests will demonstrate how effective the collection and processing of data is. For this stage, it is very important that the data created by the test users gives a fair reflection of a number of diverse areas so that the limits of the capabilities of the software can be tested, showing how successful the process has been.

Bibliography

Abiodun, O.I. *et al.* (2019) 'Comprehensive Review of Artificial Neural Network Applications to Pattern Recognition', *IEEE Access*, 7, pp. 158820–158846. Available at: https://doi.org/10.1109/ACCESS.2019.2945545.

Ahn, J., Kim, K. and Proctor, R.W. (2018) 'Comparison of Mobile Web Browsers for Smartphones', *Journal of Computer Information Systems*, 58(1), pp. 10–18. Available at: https://doi.org/10.1080/08874417.2016.1180652.

Alharbi, J.A. *et al.* (2023) 'An Empirical Analysis of E-Governments' Cookie Interfaces in 50 Countries', *Sustainability*, 15(2), p. 1231. Available at: https://doi.org/10.3390/su15021231.

Browser History: Epic power struggles that brought us modern browsers (no date) *Mozilla*. Available at: https://www.mozilla.org/en-US/firefox/browsers/browser-history/ (Accessed: 20 November 2023).

Cookies, the GDPR, and the ePrivacy Directive (2019) *GDPR.eu*. Available at: https://gdpr.eu/cookies/ (Accessed: 21 November 2023).

Das, A. (2022) *Does Disabling JavaScript Protect You From Hackers?*, *MUO*. Available at: https://www.makeuseof.com/should-you-disable-javascript-to-protect-from-hackers/ (Accessed: 21 November 2023).

Garg, V., Jegelka, S. and Jaakkola, T. (2020) 'Generalization and Representational Limits of Graph Neural Networks', in *Proceedings of the 37th International Conference on Machine Learning*. *International Conference on Machine Learning*, PMLR, pp. 3419–3430. Available at: https://proceedings.mlr.press/v119/garg20c.html (Accessed: 23 November 2023).

Gregor, P. and Dickinson, A. (2007) 'Cognitive difficulties and access to information systems: an interaction design perspective', *Universal Access in the Information Society*, 5(4), pp. 393–400. Available at: https://doi.org/10.1007/s10209-006-0064-6.

How do we comply with the cookie rules? (2023). ICO. Available at: https://ico.org.uk/for-organisations/direct-marketing-and-privacy-and-electronic-communications/guide-to-pecr/guidance-on-the-use-of-cookies-and-similar-technologies/how-do-we-comply-with-the-cookie-rules/ (Accessed: 21 November 2023).

Hr, M.G. *et al.* (2020) 'Development of anti-phishing browser based on random forest and rule of extraction framework', *Cybersecurity*, 3(1), pp. 1–14. Available at: https://doi.org/10.1186/s42400-020-00059-1.

Klopper, R., Gruner, S. and Kourie, D.G. (2007) 'Assessment of a framework to compare software development methodologies', in *Proceedings of the 2007 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries*. *SAICSIT '07: 2007 Annual Conference of the South African Institute of Computer Scientists and Information Technologists*, Port Elizabeth South Africa: ACM, pp. 56–65. Available at: https://doi.org/10.1145/1292491.1292498.

Kumar, S. and Sharma, R.R. (2015) 'Empirical Analysis of Unethical Practice of Cookies in E-Marketing', *Abhigyan*, 33(3), pp. 42–56.

Kyriakoullis, L. and Zaphiris, P. (2016) ‘Culture and HCI: a review of recent cultural studies in HCI and social networks’, *Universal Access in the Information Society*, 15(4), pp. 629–642. Available at: https://doi.org/10.1007/s10209-015-0445-9.

Leau, Y.B. *et al.* (no date) ‘Software Development Life Cycle AGILE vs Traditional Approaches’.

Li, Z. *et al.* (2022) ‘A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects’, *IEEE Transactions on Neural Networks and Learning Systems*, 33(12), pp. 6999–7019. Available at: https://doi.org/10.1109/TNNLS.2021.3084827.

Liu, C.-H., Chen, W.-K. and Sun, C.-C. (2020) ‘GUIDE: an interactive and incremental approach for crawling Web applications’, *The Journal of Supercomputing*, 76(3), pp. 1562–1584. Available at: https://doi.org/10.1007/s11227-018-2335-4.

Livingstone, D.J., Manallack, D.T. and Tetko, I.V. (1997) ‘Data modelling with neural networks: Advantages and limitations’, *Journal of Computer-Aided Molecular Design*, 11(2), pp. 135–142. Available at: https://doi.org/10.1023/A:1008074223811.

Lock, R. (no date) ‘Agile Software Development’.

Marcus, A. and Gould, E.W. (2000) ‘Crosscurrents: cultural dimensions and global Web user-interface design’, *Interactions*, 7(4), pp. 32–46. Available at: https://doi.org/10.1145/345190.345238.

Mazher, N., Ashraf, I. and Altaf, A. (2013) ‘Which web browser work best for detecting phishing’, in *2013 5th International Conference on Information and Communication Technologies*. *2013 5th International Conference on Information and Communication Technologies*, pp. 1–5. Available at: https://doi.org/10.1109/ICICT.2013.6732784.

Mishra, P. *et al.* (2020) ‘New data preprocessing trends based on ensemble of multiple preprocessing techniques’, *TrAC Trends in Analytical Chemistry*, 132, p. 116045. Available at: https://doi.org/10.1016/j.trac.2020.116045.

Oxford Languages and Google - English | Oxford Languages (no date). Available at: https://languages.oup.com/google-dictionary-en/ (Accessed: 22 November 2023).

Pusara, M. and Brodley, C.E. (2004) ‘User re-authentication via mouse movements’, in *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*. New York, NY, USA: Association for Computing Machinery (VizSEC/DMSEC ’04), pp. 1–8. Available at: https://doi.org/10.1145/1029208.1029210.

Qiao, Y. *et al.* (2020) ‘Siamese Neural Networks for User Identity Linkage Through Web Browsing’, *IEEE Transactions on Neural Networks and Learning Systems*, 31(8), pp. 2741–2751. Available at: https://doi.org/10.1109/TNNLS.2019.2929575.

Recital 30 - Online identifiers for profiling and identification (2018) *GDPR.eu*. Available at: https://gdpr.eu/recital-30-online-identifiers-for-profiling-and-identification/ (Accessed: 21 November 2023).

Sa-nga-ngam, P. and Kiattisin, S. (2020) ‘Increasing Web Accessibility Through a Personalized Web Browser for Older Adults’, *Wireless Personal Communications*, 115(4), pp. 3235–3259. Available at: https://doi.org/10.1007/s11277-020-07220-6.

Shakir M. Abass, K.J. (2019) ‘Development History Of The World Wide Web’. Int. J. Sci. Technol. Res, 8(9), pp.75-79. Available at: https://www.researchgate.net/profile/Karwan-Jacksi/publication/336073851_Development_History_Of_The_World_Wide_Web/links/5d8d1f8f92851c33e94064cb/Development-History-Of-The-World-Wide-Web.pdf.

Su, Q.-Y. and Adams, C. (2005) ‘Will B2C e-commerce developed in one cultural environment be suitable for another culture: a cross-cultural study between amazon.co.uk (UK) and dangdang.com (China)’, in *Proceedings of the 7th international conference on Electronic commerce*. New York, NY, USA: Association for Computing Machinery (ICEC ’05), pp. 236–243. Available at: https://doi.org/10.1145/1089551.1089598.

Thivaharan., S., Srivatsun., G. and Sarathambekai., S. (2020) ‘A Survey on Python Libraries Used for Social Media Content Scraping’, in *2020 International Conference on Smart Electronics and Communication (ICOSEC)*. *2020 International Conference on Smart Electronics and Communication (ICOSEC)*, pp. 361–366. Available at: https://doi.org/10.1109/ICOSEC49089.2020.9215357.

Vidas, T. *et al.* (2013) ‘QRishing: The Susceptibility of Smartphone Users to QR Code Phishing Attacks’, in *Financial Cryptography and Data Security*. *International Conference on Financial Cryptography and Data Security*, Springer, Berlin, Heidelberg, pp. 52–69. Available at: https://doi.org/10.1007/978-3-642-41320-9_4.

Virvilis, N. *et al.* (2015) ‘Security Busters: Web browser security vs. rogue sites’, *Computers & Security*, 52, pp. 90–105. Available at: https://doi.org/10.1016/j.cose.2015.04.009.

Vizer, L.M. (2009) ‘Detecting cognitive and physical stress through typing behavior’, in *CHI ’09 Extended Abstracts on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery (CHI EA ’09), pp. 3113–3116. Available at: https://doi.org/10.1145/1520340.1520440.

What are the limits of deep learning? (no date). Available at: https://doi.org/10.1073/pnas.1821594116.

What Is Web 3.0 (Web3 definition)? (no date) *What Is Web 3.0 (Web3 definition)?* Available at: https://www.avast.com/c-web-3-0 (Accessed: 20 November 2023).

Yu, L., Li, Y. and Zeng, Q. (2021) ‘Design of topic Web crawler based on improved PageRank algorithm’, *Journal of Physics: Conference Series*, 1754(1), p. 012210. Available at: https://doi.org/10.1088/1742-6596/1754/1/012210.