Developing a web browser and Chrome extension with on the fly filtering and parent-child monitoring using machine learning algorithms, Node.js and Nw.js

Institiúid Teicneolaíochta Chorcaí
Cork Institute of Technology

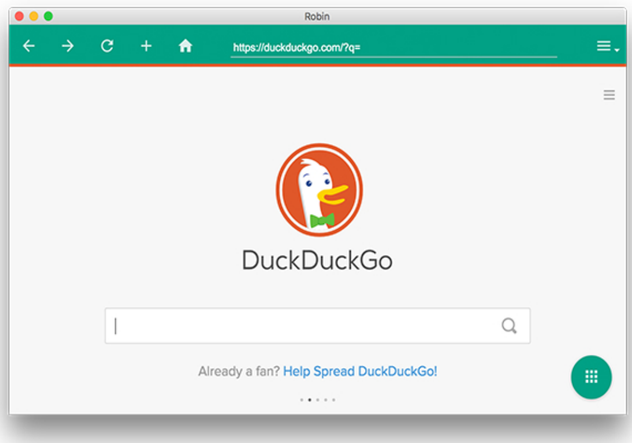
Robert James Gabriel
Dr. Donna O'Shea

Introduction

• The objective of the project is to develop a web browser using Node.js Webkit with a built in filtering system to block websites/content that the parent wouldn't want their child to access while allowing them to monitor their child's activity from a chrome extension.

• There are several parts to this project. A parser/scraper, a verb checker as the application will be collecting data on the web page along with explicit and positive words for training the Naive Bayes classifier algorithm which will determine if a web page is negative or positive.
• As an existing problem with filtering/monitoring systems is false positives, where sites that are safe are blocked when they shouldn't be.

• Example, most sexual education sites are blocked because they use sexual words.
• The goal of this should help with the false positive effect of sexual education websites.



Web Browser

Web Browser & Chrome Extension

Features
Refresh page,
Tabs
Go Back/Forword,
Go Home,
Search the web,
Change the theme

Features
Set Blacklist sites
Set Whitelist Sites
Personal settings
Montor what users are on
Stop access to the web.

Built using Angular.js, Node, npm, Github, Firebase,Gulp Nw.js and less.

Design: Material Design which makes use of grid-based layouts,animations and transitions, padding, and depth effects such as lighting and shadows.

Naive Bayes classifier Algorithm

The Naive Bayes classifier algorithm uses training data from the web browser's parsing and scraping of text on a website.

Each word is classified into either negative or positive and stored in the database .
Any new words are stored in firebase and used for training the algorithm.

So when a child goes onto a new website, it will parse the text and place the words into the Naive Bayes classifier algorithm and will return either if its negative or positive webpage.

Example of training Data

Good Words:["the","of","and","to","a","in""]
Negative Words: ["4r5e","5h1t","5hit","a55","ar5e"]

Likelihood
Class Prior Probability

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

Posterior Probability
Predictor Prior Probability

Application Design

• The application is built using Angular.js MVC pattern written in Node.

• It is split into the following individual packages

A Parser
Web Scraper
Verb Checker
Firebase
WorkHorse (Works on large sets of data over time)
Native Baylor Algorithm (Fliteirng system)

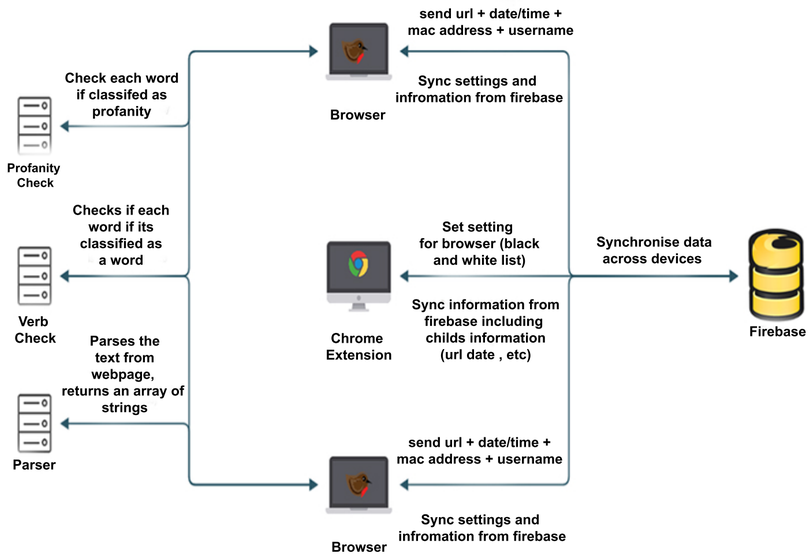


Figure 3.1

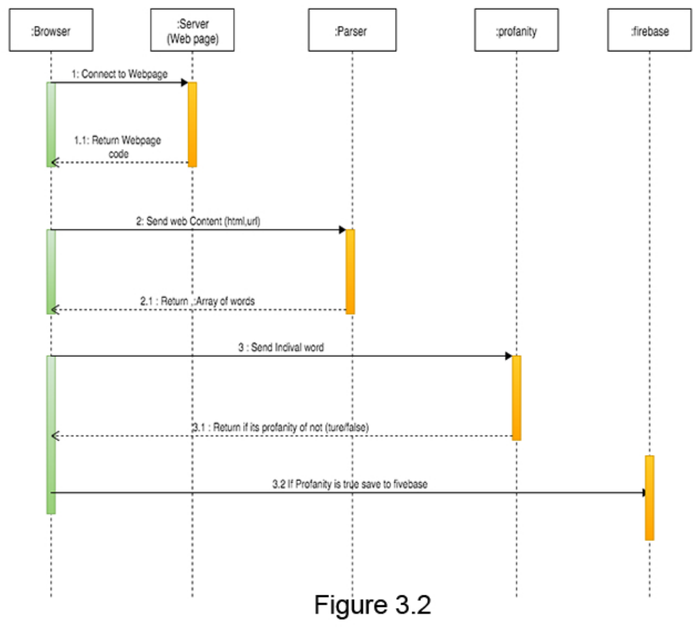


Figure 3.2

Tests & Results

Stress testing/Load testing
There were several problems and boot necking/freezing with loading the 14,000 datasets. The solution came with loading 1,000 datasets every 5 seconds removing strain on the application and decreasing training time.

Performance testing: There was 13 version of the application. Each version was monitored for performance issues along with optimizing code & resources.
See **Figures 1.1 and 1.2** for lower cpu usage and ram useage over revesions.

Regression testing was used to ensure any updates to code didn't break anything; this was automated using gulp.js.

Accuracy testing: The accuracy of the Naive Bayes classifier was 50% with 50/50 negative positive words. After analyzing the data, having the positive words to a weight of 75/25 gave results of **70% - 80%.** See **Figures 2.1 & 2.2** for before/after

Usability testing: used in the testing (a-b testing) to evaluate the design and revisions were done for the icons and chrome extension layout.

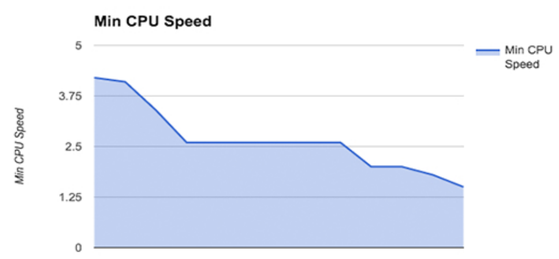


Figure 1.1

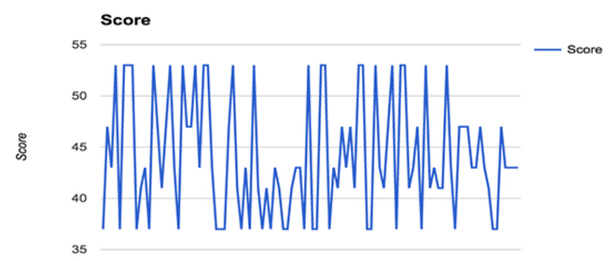


Figure 2.1

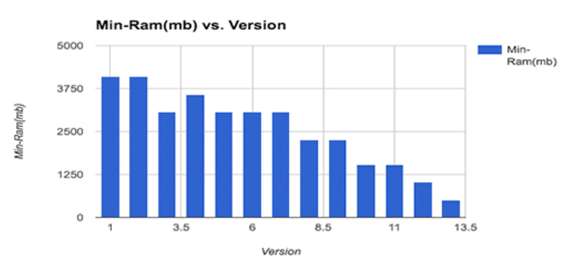


Figure 1.2

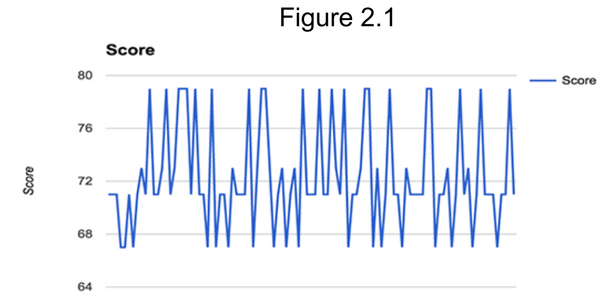


Figure 2.2

Conclusion

•The use of the MVC patttern is deal for web apps that need to be scalable and use individual parts

• Halving the **positive words over the negative words was achieving a result of 70% to 80%** in classifying adult sites as negative and others as positive.

• The reason is that the positive and negative words both appear in adult sites.
• The use of a workhorse for processing large data is amazing to relieve strain

•There is room for further study into this area of word analytics for websites.

Also with Nw.js, Gulp, Npm and Mateiral Design

Acknowledgments: The author would like to acknowledge the support of Dr. Donna O'Shea & Dr. Ted Scully, Reddit for all the late night procrastination.