Secured Client-Side Content Filtering Using Machine Learning Algorithms

M. V. Ramana Murthy¹, Ermias Bayu Woldegiyorgis² and M. Pavan Kumar³

¹School of Computer Science, Osmania University, Amberpet, Hyderabad – 500007, Telangana, India; mv_rm@rediffmail.com
²Department of Mathematics, Osmania University, Amberpet, Hyderabad – 500007, Telangana, India;
³Department of Electronics and Communication, Osmania University, Amberpet, Hyderabad – 500007, Telangana, India;

Abstract

Objectives: The sheer amount of content available in cloud service providers (News, Social media, etc.) poses a fundamental problem for users: The excess amount of diverse information available for public access without regard to age, sex or other classifiers. Methods/Statistical analysis: Ability to filter and control this content while maintaining anonymity is a challenge which isn’t addressed properly, yet. Findings: Previous Solutions proposed to tackle this need integration on either server-side or client-side. However, these solutions seem to have two major drawbacks the ability to maintain client anonymity from the content provider and the ability to integrate server-side filtering with client side filtering. Application/Improvements: In this paper a system is proposed that integrates secured content classification and content filtering based on trained data. Security is achieved by using the identity of the user generated from the initial contact point.

Keywords: Jason Web Token (JWT), Machine Learning, Naive Bayes, Tiny Encryption Algorithm (TEA)

1. Introduction

We live in an age of big data. With hundreds of millions of people spending countless hours on social media to share, communicate, connect, interact, and create user-generated data at an unprecedented rate, social media has become one unique source of big data. People share what they like, what they don’t like, pictures, movies, and many other things. People of all ages and gender are exposed to this content without any filter or regard to their psychological as well as moral impacts. In the past, the ability to filter this content to prevent exposure to unnecessary, vulgar, harmful and unwanted information has been explored deeply. What has been missing so far is the ability to filter this content while maintaining anonymity and privacy from the content provider?

We have found that most systems and proposed mechanisms that use machine learning Algorithms lack the security and privacy feature required by most users while filtering content. In a world where security and privacy have become a billion-dollar business the ability to know ones like and dislikes is not something to be seen lightly. Systems proposed previously expose the filtering requirements of users to content providers (i.e. Google, Facebook, Quora etc.) and this is mainly due to the fact that both the content and the filtering systems are provided by the content providers themselves.

The aim of this work is to propose a secured client side strategy that can enable filtering of contents available on content based and sharing sites and applications. This is achieved by moving the classifying algorithm from the server side to client side (i.e. the browser) and enabling encrypted access to third party and user clients whenever necessary. Moving the filtering and classification algorithms to the client side can provide a secure and fast classification by shielding the users feeling and filtering needs from the content owner.
To show our proposed strategies we have built a Proof of concept system that integrates the filtering and security feature of the proposed strategy which is implemented as a web browser extension used to filter content from Facebook.com posts in a secure and privacy-preserving manner from the client side.

2. Architecture

The architecture of the system we propose is depicted below in Figure 1. As shown, the system will have 3 main layers implemented in different environments and technologies. In The First Layer classification model generator will handle all the necessary training data and dataset related issues with this regard this layer is the only layer implemented on server side to avoid redundancy and to minimize cost and efficacy factors it also has a security feature which is used to provide encryption feature for the generated data and authentication feature for users. This specific layer is the one responsible for generating a classification model that will be stored on the client side for later use. The second layer which is the main component of the overall filtering system is the filtering application which is responsible for extracting text from content provider /sharing sites, in addition, it also integrates the short text classification algorithm which will utilize the classification model. Users of the system Interact through the third layer of the architecture which is provided as an extension the browser which is a Graphical User Interface (GUI) based browser extension.

Steps of operation on the system is to filter unwanted content is explained here step by step and is pointed below on Figure 2. Operations involved for filtering certain content.

1. User provides filtering requirements through the extension popup — 1
2. Using the available training set the classification model generator will load and train the algorithm and generate the model in a JSON format. — 2
3. The classification model will be stored and made available in an encrypted format to the user using the information (which is provided during initial registration) as a key. — 3 - 5
4. Whenever a certain content based Paige is loaded into the browser the filtering apps loads the Document Object Model (DOM) and analyse the DOM to find content trees, and then the filtering module sends an array of short texts to the text classification module. — 1, 5
5. The text classification module load and decrypts the classification model. — 6
6. The classification model loads the classification (training) model after encrypting it to classify the extracted texts from the DOM and sends the result to the filtering module. — 7, 8, 9,
7. The filtering module loads the filtering requirement from local storage and compares it to the results from the classification algorithm after which it will restructure the DOM on the content provider’s page by deleting certain nodes from the DOM accordingly. — 10

3. Classification Model Generator

This layer of the system is the part with will mainly be running on the back end side of the system. This layer will be responsible for training and generate different classification models used for text classification while filtering content.
This layer will have mainly of two modules the training algorithm which will consume the dataset provided to generate an acceptable classification model and the security module.

3.1 Data Set
A combination of two different datasets was used as the testing and training data sets to generate and test the trained data model. (Reuters-21578 dataset and a custom generated dataset from Wikipedia texts). For testing purposes of this paper we generated and used two classes (Politics and Music)

3.1.1 Training Data Set
This data set has a class that has labels in it. The class labels are known. Classification model is generated from the training data set which in future is used to predict the class of unknown contents.

3.1.2 Testing Data Set
The data set which has similar description like training dataset but the values of attributes are not known. Values of these attributes are predicted by the previous observations of data. So we start using machine learning algorithm for finding the patterns in the data and then these attributes are associated with the value of the class. Testing data set is used to determine the accuracy of the model. Model is applied on testing data set.

3.2 Training Algorithm
Supervised learning --- the algorithm generates a function that maps inputs to desired outputs. One standard formulation of the supervised learning task is the classification problem: the learner is required to learn (to approximate the behavior of) a function which maps a vector into input-output examples of the function.

The machine learning algorithm we decided to test our strategy is Naive Bayes classifier algorithm which is a conditional probability model. Which is easy to build, understand and debug, thus making it good to show and prove our point without spending too much time on the intricacies of the Machine learning algorithm. As expected the algorithm will have to main parts: the training part and the classifying part.

In addition to the training and classification algorithm, we have also used the implementation of The Porter stemming algorithm (or 'Porter stemmer') which is a process for removing the commoner morphological and in flexional endings from words in English. Its main use is as part of a term normalization process that is usually done when setting up Information Retrieval systems.

According to our proposed strategy, we will be separating the training and the classification algorithms to the server side and the client side consecutively. Thus we will use the Training algorithm in Naive byes to train and save the trained data model as a JSON object on the server.

3.3 Security
We implemented a two-way security feature to secure the classification model and filtering related information from both the server and client side. The steps implemented in this layer are:

1. Generate a Jason Web Tokens(JWT) for every user in the time of registration and download
2. Use the generated JWT as a password (key) to encrypt the generated JSON file (classification model) while sending it to individual clients for access from the client side.

We used the Tiny encryption algorithm to encrypt and decrypt content both form the client and server side.

The Tiny Encryption Algorithm which one of the fastest and most efficient cryptographic algorithms is a feistel based block cipher which was developed by David Wheeler and Roger Needham.

4. Filtering Application Layer
This is the main layer of the system used to classify and filter content from the provider site. This layer is completely implemented and hosted on the browser as an extension client side app.

4.1 Filtering Module
This module in the filtering layer is the module that handles DOM processing algorithms that go through the DOM to identify and college's content that needs to be filtered are implemented within this module.

The main operations inside this module will be DOM manipulation which includes:
Secured Client-Side Content Filtering Using Machine Learning Algorithms

- Loading and collecting DOM content once the document finishes loading from the browser.
- Manipulating DOM content to delete /hide content that satisfies categories set by the user.

4.2 Short Text Classification

Short text classification is implemented in the client side of the application which will be supported by loading the classification model which is loaded to the classification algorithm.

As explained in the section III-C Naive Bayes algorithm will be used both to generate the classification model and classify content.

The only difference from tradition implementation of the algorithm is that this will be implemented on the client side while the training data and algorithm will be available on the server side. The classification algorithm will be able to classify content based on the generated classification model (during the training of the algorithm).

5. Filters Wall

The Filter wall layer contains mostly GUI for collecting data from the user of the system and configuring user settings to the user.

5.1 Web Interface

The web interface of the application is the initial point of contact with the user. The user will be able to download and configure its settings for securing any future communication to the server from the application. The user will be assigned a specific JWT which is stored on the server along with any user data used to generate it and will be used as a key to encrypt data and the classification model whenever the user makes contact with the system. The user will also be able to view and analyze details of blocked contents from a dashboard provide by the web interface of the application.

5.2 Extension Interface

As a proof of concept, we implemented a Chrome extension to show the applicability of our theory to the specify problem. The extension interface will be used to collect and pass user filtering requirements from the client side and interact with the user natively in the browser.

6. Conclusion

Using the proposed strategy, we were able to provide a filtering mechanism from the client side that is more secure, fast and reliable for the users of cloud content providers.

7. Future Work

In this work, we were mainly focused in devising a secured way to filter content from the client side that is intelligent and more accurate. In future works, a deepest and elaborate test need to be done to see the strength of the security technique and the ability of the browser to respond and process to huge classification models stored locally on local storage facility provided by it.

8. Acknowledgment

The authors would like to thank the following for helping and providing the required facilities and funding for the study: Osmania University · Department of Mathematics, Indian Council for Cultural Relations, Information Network Security Agency.

9. References

1. Zafarani R, Abbasi MA, Liu H. Social media mining an introduction. Cambridge University Press; 2014 Jan.
2. Chau M, Chen H. A machine learning approach to web page filtering using content and structure analysis. Decision Support Systems. 2008 Jan; 44(2):482–94.
3. Adomavicius A, Tuzhilin G. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. Institute of Electrical and Electronics Engineers (IEEE) Transaction on Knowledge and Data Engineering. 2005 Jun; 17(6):734–49.
4. Chau M, Chen H. A machine learning approach to web page filtering using content and structure analysis. Decision Support Systems. 2008 Jan; 44(2):482–94.
5. Mooney RJ, Roy L. Content-based book recommending using learning for text categorization. In the Proceeding of the Fifth Association for Computing Machinery (ACM) Conference in Digital Libraries; 2000 Jun. p. 195–204.
6. Vanetti M, Binaghi E, Ferrari E, Carminati B, Carullo M. A system to filter unwanted messages from OSN user walls. Institute of Electrical and Electronics Engineers (IEEE) Transactions on Knowledge and Data Engineering. 2013 Feb; 25(2):285–97.
7. Reuters-21578 Text Categorization Collection Data Set [Internet]. 1987 [cited 1987]. Available from: https://archive.ics.uci.edu/ml/datasets/Reuters-21578+Text+Categorization+Collection
8. Porter MF. An algorithm for suffix stripping. Program: Electronic Library and Information Systems. 2006; 40(3):211–8.
9. Implementation of the Tiny Encryption Algorithm [Internet]. 2015 [cited 2015]. Available from: https://github.com/chrisveness/crypto/blob/master/tea-block.js
10. Wheeler DJ, Needham RM. TEA, a tiny encryption algorithm. In: Preneel B editor. International Workshop on Fast Software Encryption. Lecture Notes in Computer Science, Springer, Heidelberg. 1994 Dec 14–16; 1008:363–6.