Creating a Digital Fingerprint From Web Browsing History Alone

Stephen Santise
807 Union St.
Union College
Schenectady, New York 12308
(516) 398-0142
santises@union.edu
ssantise@gmail.com

Aaron G. Cass
220 Steinmetz Hall
Department of Computer Science
Union College
Schenectady, New York 12308
(518) 388-8051
cassa@union.edu

ABSTRACT

Humans are unique creatures. Everything we do is slightly different from everyone else. Even though many times these differences are so minute they are not even noticed. This is exactly the same for web browsing. The way that each human browses the web is unique. Each websites we visit as well as the order in which we visit them is unique.

Now wouldn't be nice if this uniqueness was not just overlooked and was actually used to benefit the user's browsing experience. In this research we compare different representations of browsing histories to find which one can best be used to represent this uniqueness. Then by using machine learning algorithms this research will attempt to create a fingerprint from which future web-browsing could be used to identify the user who created it.

1. INTRODUCTION

As technology progresses, society finds itself spending more and more time online surfing the web. Just like any other activity the way we surf the web is unique to each and everyone of us. This uniqueness is what my research wishes to exploit.

Imagine that you are working for the CIA and are currently tracking fifteen different terrorists. You have an extensive dossier on each terrorist containing all different types of information, including months of browsing history from each terrorist. Now while tracking these terrorists the government raids a terrorist compound and among the evidence there is a laptop computer. Unfortunately you have no idea who the laptop belongs to. Every reasonable way of determining who owns the laptop is exhausted yet you are still unable to identify who owns it. Leaving only one option; to run the laptops browsing history through my program. By comparing this laptops browsing history against the previously collected histories you will be able to accurately identify which of the terrorists (if any of them) this laptop belongs to.

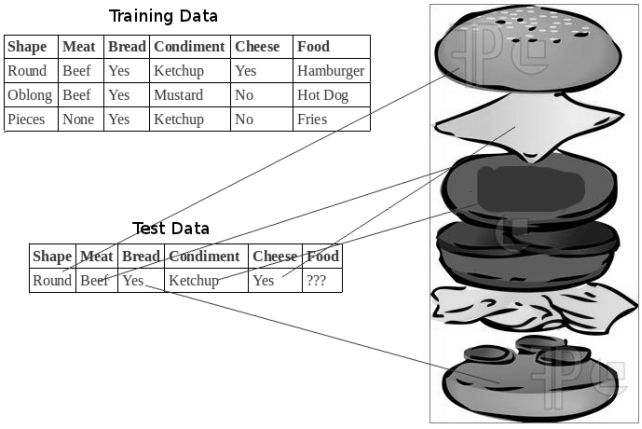

Figure 1. Classification Example

However to correctly identify a user based on their web-browsing history alone is no easy task. The absolute easiest way to to identify a user would be a simple data look up that would take the user's browsing history represented as an ordered list of web-pages as an input. Then perform a simple search to determine if any of the previously saved histories have the exact same web-pages visited as the user in question returning the user who matches. This however is a terrible way of identifying a user because the chances that the user's new history being exactly the same as a previously recorded history are extremely low.

Clearly this problem can not be solved with a simple search so we must utilize techniques from other fields of computer science. Mainly the field of classification or data mining. Classification is an extension of the machine learning field, a simple explanation is it takes a set of training data which contains sets of known features with specific variables that correspond to the attributes. To better understand this concept refer to figure 1. In this example the program is trying to identify what the unknown food is based on its ingredients. First the program is trained by storing known examples of foods based on their ingredients called features or attributes. Each type of food may have different variables for each attribute but they all carry information for each attribute. The test data is then passed through a classification algorithm which uses the previously trained data to classify the new test data and return the type of food that the test data most likely represents.[13]

Table 1. Basic Classifier

Page1	Page2	Page3	Every Web-page Ever	PageN	User
True	False	True	…	True	Bob
False	False	False	…	True	Joe
True	True	False	…	False	Tim

The most basic way to classify (or in this case identify) the user based on their web-browsing history can be seen in table 1. The first step is to create a library of every web-page on the Internet. Each web-page will be an attribute and a boolean will be the variable to determine if the web-page has been visited or not. Obviously this will be quite a large list but every user must have the same number of attributes to represent them in order for the

classifier to work. Next the classifier would be trained on known users and their corresponding browser histories. Then when the test data from the unknown user is compared to the trained data the classifier will attempt to identify the user by finding the similarities and dissimilarities between the test data and the trained data. This again however is clearly not the best way to represent the data because having a instance of every single web-page is impractical.

Which leads to the question: What is the best way to represent a user's browsing history to achieve the best identification accuracy?

2. RELATED WORK

2.1 Device Fingerprinting

As modern technology progresses privacy becomes a bigger and bigger issue. Even modern web browsers are vulnerable to being fingerprinted.

Peter Eckersley's research [11] investigates exactly how vulnerable modern web browsers are to "device fingerprinting." To create a fingerprint of the browser, version and configuration information that is transmitted to websites will have to be collected. From this information many different variables can be identified such as screen resolution, plug-ins installed, cookies identified, installed system fonts determined, etc. By identifying and recording this data a fingerprint can be made for that specific computer. Then when that computer visits the site again by using specific algorithms that handle updated fingerprints (because they change between sessions due to updates and continued browsing) the website will be able to identify the user. This research is a bit different than my research because in order to make a fingerprint it requires finding out about the computer itself not the web-browsing history specifically. It does however deal with history to some degree through its use of cookies which are received from previously visited websites. Most importantly though, Eckersley's research showed that based on the variables they used "that if we pick a browser at random, at best we expect that only one in 286,777 other browsers will share its fingerprint."

[3] For years now Google's Internet advertising has been keeping records on anyone signed into a Google account by tracking users based on websites they visit that contain a Google Ad. Conti's research however suggests that is only a matter of time before we reach the "fingerprinting threshold." The "fingerprinting threshold" is defined as the amount of data that a information service provider would have to collect in order to correctly identify the user based on a saved fingerprint. Similar to Eckersley's research though the information that would be needed is beyond that of just browsing history. Also this research looks to make a specific fingerprint that would be the same or close to the same each time a user logs on. However though since this research Google has begun to use browsing history specifically in order to better recommend advertisements [14] so its only a small logical step before it could become a part of the device fingerprint.

2.2 History Sniffing

History Sniffing or History Hijacking is currently used by many different websites to secretly record information about the users who visit their website. In both Jakobsson's [7] and Hall's [8] research they show that websites are currently keeping records of what websites were visited prior to itself and in many cases transmitting the data across its network. Basically what happens is that the website exploits the :isVisited function in CSS or it exploits a similar function in Javascript. By using these functions and a simple loop, code can be written to iterate through a set of websites and determine if the user has browsed them. However this will only return which websites have been visited within the current browsing session. Even though there is no evidence to suggest that websites are currently identifying specific users based on this data, in the future this could change. Especially as technology progresses and greater sets of websites can be iterated through and if stronger correlations are found between users and patterns in their browsing history.

2.3 Web Search Personalization

Everyday people all across the globe go online and use search engines to find out information about their query. There are many times when multiple users will enter the same query yet expect to find different results.

In Rastegari's research [11] they use web browsing history along with an Artificial Immune System to personalize the search results. In this research the user's browsing history is used to create a profile of the user that will then be used by the search engine to tailor the returned results to the specific user. However this research does not stop at just browsing history but goes on step further and parses through the websites to find out more about them. What is important to note is that user profile is similar to the fingerprint discussed in this research and which is then identified by the search engine.

Kazunari's research [6] although very similar to Rastegari's research in the way that user profile is created but differs in the way that the search engine is personalized. Kazunari's research records a library of user profiles then finds similarities between the users to create a neighborhood of similar users. Finding similarities between the two users is very much the same as finding similarities between trained data and test data.

2.4 Text Classification/Categorization

Text classification is the act of classifying a natural language text based on a predefined set of categories. Text classification works very much the same way as regular classification, first the classifier is trained with predefined training data then the classifier is run on the text to identify what classification the text falls under. [12] Although I will not be classifying any text in my research, the principle of classification is the same also data sets of text are similar to that of my data. When dealing with text there are many different features (i.e. the words) just like in my data (i.e. the websites) and because of this many of the same ideas apply.

Research has also been done to determine the efficacy of representing the text to the machine learning algorithm by the sequence it was written in. In Cavnar's research [1] they show how the sequence of the letters in the text could be used to classify the text as a whole. In this experiment Cavnar takes natural language text and breaks down into N-grams. A N-gram in this case is the a slice of N-characters from a word. For this example of N-grams the word "TEXT" would be composed of these tri-grams: _TE, TEX, EXT, XT_,T__. Then to categorize the text the test data was compared to training data that was trained using N-grams. This research showed that even though the program was not actually understanding any of the words it was still able to effectively categorize the text.

2.5 Weka

Weka (Waikato Environment for Knowledge Analysis) is a suite of machine learning algorithms that can be used for various data mining tasks. Weka can be used directly with a .arff database file through the GUI or you can implement the algorithms using predefined libraries in Java. Weka is open source and licensed under the GNU General Public License. Weka is popular within the machine learning community because it provides clean and easy ways to implement most classifiers.

3. GENERAL APPROACH

In order to identify a user based on their web-browsing history there are a series of steps that must be taken in order to correctly identify the user.

3.1 Format Data

The first step is to obtain a dataset of web browsing history and convert it from its original format into a usable format. In this case an ordered list of strings containing the http:// addresses visited in the history will be a good start.

3.2 Different ways to Represent Data

Now that the data is in a usable format we must decide how we want to represent the data for each user to the classifier. We have already seen the most basic way to represent the data in Table 1. To expand from the basic representation there are two different aspects of the data that can be manipulated in order to represent the data differently. These aspects are specificity and sequence.

3.2.1 Specificity

There are two different levels of specificity that we will be using to determine which works best. The two levels are web-page and website. A web-page would be the exact http:// address that appears in the history already (i.e. http://www.union.edu/academic/index.php). A website would be a broader category consisting of just the site name (i.e. union.edu).

The initial motivation comes from a simple analysis of my own browsing history. I noticed that within my own browsing history that due to the high specificity of each web-page that there was a very small amount of times where I revisited the exact same web-page while there was quite large amount of times where I visited the same website. By testing both levels of specificity the experiment will be able to determine if it is more important that a user visited the same web-page a few times or if it's more important that the user visited the same website a bunch of times.

Site1	Site2	Site3	Every Website ever	Site100000	User
Yes	No	Yes	...	Yes	Bob
No	No	No	...	Yes	Joe
Yes	Yes	No	...	No	Tim

Visited 1st	Visited 2nd	Visited 3rd	User
Site2	Site3	Site5	Bob
Site4	Site2	Site10	Bob
Site6	Site7	Site1	Bob
Site8	Site54	Site10	Joe
Site92	Site122	Site35	Joe

Figure 2. Change in Representation of Data

By looking back to Table 1 you can see how the feature webpage1 … webpageN could easily be changed to website1 … websiteN or siteType1... siteTypeN. By making these changes you are drastically cutting down on the size of the feature vector associated with each user because as explained before there are far less websites the web-pages.

3.2.2 Sequencing

Through analysis of my browsing history as well as conversations with my peers about there browsing habits I began to realize that the order in which websites are visited in may also be important. For example certain people I talked to explained that when they went online they would regularly visit the same websites in the same order. This could be very important information when trying to differentiate between two users who visit many of the same websites.

In order to determine if important patterns can be found in the sequence in which the pages were visited I will use N-grams to represent the data. For example if a user had a browser history that looked like [site1, site5, site3, site4, site1] Then this history would contain the bi-grams [(site1,site5), (site5,site3), (site3,site4), (site4,site1)] and the tri-grams [(site1,site5,site3), (site5,site3,site4), (site3,site4,site1)]. By breaking the data up into sequences it drastically changes the entire representation of the data. This change can be seen in Figure 2, where the first table is how the data was previously represented and the second table being how the data will now be represented if the data was represented by trigrams.

By changing the way that we represent the data to include sequencing we do not save all that much computational speed because the value of each feature instead of being binary now becomes a variable containing every thing all of the collected users histories. However this change does allow for additional data to represented at no additional computational cost, which is the sequencing of websites visited. In the original representation sequencing was completely neglected and was only showing if the user did or did not visit a website.

3.2.3 Creating Training and Testing Sets

During the process of creating each dataset representation one Training set will be made for each representation consisting of the first 80% of each user's browsing history. The training sets simulate the database of prerecorded browsing histories that would be needed to identify a unknown browsing history.

Then for each training set a testing set will be created for each user that contains the last 20% of the user's browsing history. Each test set simulates the unknown browsing history that we wish to identify.

3.3 Training a Classifier and Testing it's Accuracy as an Identifier

In order to accomplish the task of identifying a user from a unknown browsing history, we need a way to train the identifier on browsing histories which we know the creator of. To do this we will employ a type of machine learning algorithm known as a classifier.

The training sets created in the previous step will now be used to train the classifier and then each user's corresponding test set will be used to test the classifier. Each training set's and corresponding test sets' data has been formatted in the exact same way so that each user has the same features in their feature vector. There are many different types of classifiers each with their own positives and negatives. For this reason, this research will compare the results of the K-nearest Neighbor classifier, Naïve Bayes classifier, the Random Tree classifier, the Random Forest Classifier, and the ZeroR classifier in order to determine which will work best for the purpose of identification. Each classifier is similar in that it employs a learning algorithm to identify a model that best fits the relationship between the features and the class label but different in the way it accomplishes that task. [13] Each model is then saved so that it can be applied to the test set.

Next each classifier will apply the model to each corresponding test set (which contains the untrained data from each individual user). The classifier will then predict a user classification for each instance within the test set. Each one of these classification on their own are not that useful to us because we are not trying to predict the creator of each specific sequence of web-pages or websites but rather we wish to identify which user created the unknown browsing history which would contain a set of multiple sequences. To identify a user who created the unknown browsing history each instance's classification is recorded. Then whichever user classification is predicted the most times is determined to be the user who created the history. We then record if the testing set was predicted correctly or not.

3.4 Evaluating the Identifier

In order to determine how well the identifier performed there are two different metrics used. First for each training set that is trained on each classifier is ranked by the percentage of correct user identifications that it predicted.

Then when there are multiple identifiers with the same percentage of correct identifications a confusion matrix can be created to show the counts of correct and incorrect classifications of the users.

Table 2. Confusion Matrix

		Predicted User		
		User1	User...	UserN
Actual User	User1	*x*	...	y
	User...	...	...	...
	UserN	z	...	b

Each variable (x,y,z,b) represents number of times the user from that row was classified as the user from that column. The confusion matrix does a great job of displaying the information of how well the classifier did it does not provide a summary of how the classifier did overall. In order to summarize this information in a single number a performance metric can be used. In order to find the accuracy of the classifier all that needs to be done is a simple division of the total number of correct predictions by the total number of predictions.

4. METHOD/EXPERIMENT

My research is not trying to just identify a user based on web-browsing history but rather trying to find the best way to represent the browsing history to achieve the best identification accuracy. To complete this task I have formulated an experiment.

4.1 Obtaining Data

The very first step as it is most research is obtaining data. After an exhaustive search for a dataset of browsing history online it became apparent that I would need to collect the data myself from live volunteers. First permission was obtained from the Human Subjects Review Committee to collect the data. Next through the use of flier's and a $10 compensation I was able to recruit 11 volunteer's provide with the necessary data for my research.

Once permission and volunteers was obtained I personally set each volunteer's computer up so that it would store their browsing history separate from their previously recorded history. For the purposes of this research only browsing histories created using Mozilla Firefox were used to ensure that each browsing history was initially represented the same way. This was accomplished by either installing Mozilla Firefox onto their computer or using an already installed version of the browser. Next I created a second research profile for Firefox through the use of Firefox's profile manager. Each volunteers Firefox was set to prompt the volunteer to choose which profile to use each time Firefox was run. This was done so the volunteers could choose whether their proceeding browsing history would submitted to my research or not. Thus lowering the risk of the user submitting any web-page they may of found to be too personal.

Once each volunteers' computer had been configured to save the browsing history for the experiment the user was then asked to browse the internet within the experimental profile whenever they felt comfortable for a period of one month.

After the end of the data collection period I will met back up with each volunteer and manually retrieved the history file saved in the research profile. The browser history can be found within the directory ~/mozilla/firefox/"random string"."Research" The file within this directory which contains the browsing history is labeled places.sqlite This is the file that I need. Each user was both debriefed with the Human Subjects Review Committee approved debriefing and compensated with $10.00 for their participation.

Once each of the 11 volunteers' browsing histories had been obtained I added my own browsing history file to the set. Then each file was ran through a script to randomly rename every file with a name in the format UserN where N is a unique and random number from the set of total volunteers. This served two purposes: First it changed the name of the files so they are not all called places.sqlite. Second this increases anonymity for the volunteers.

Of the 11 different browsing histories collected 3 were not used due to a insufficient amount of pages visited (each was less then 200 pages). Leaving 9 (8 volunteers plus my own) unique browsing histories to be experimented on.

4.2 Formatting the Data

places.sqlite contains much more data beyond that of just the browsing history. The first step is to extract just the browsing history from the sqlite database. To do this I will use a sql viewing GUI that will allow me to save just the browsing history in CSV (comma separated value) file. This will have to be done manually for each file. This however still contains more data then is needed.

The next step is to pass the newly created csv files into a python script that will strip out any strings that are not preceded by http:. This accomplishes two things: First it removes the extra data in the history such as search inputs; Second it removes any secure websites (i.e. websites preceded by https:) again this is done to protect the privacy of volunteers. Once the data is fully stripped it is then saved in array for each user.

4.3 Creating Different Representations of the Data

4.3.1 Converting the Browser Histories To Increase Privacy and Anonymity

In a real life situation an identifier would use the strings contained in each user's data array as is. In this research however this is not the case because in order to increase anonymity of each user we do not want to have the actual web-page string recorded.

To accomplish this first each users data array is duplicated then stripped down to contain an ordered list of strings representing each website visited. Now each user will contain two arrays, one for web-page specificity and one for web-site specificity.

Next each web-page string in each user web-page data array is stored in one grand library after checking that the page does not already exist in the library. Then each user's web-page data array is updated by changing each web-page string to a numeric value corresponding to the index of the web-page string in the grand library. The grand library is then looped over to change each value from a string to a integer equal to its current index. This same process is then repeated creating a second grand library for the website specificity

At this point every web-page string has been changed to a integer value in order to provide the volunteer's with the greatest amount of privacy and anonymity possible.

4.3.2 Creating Each Dataset That Will Be Used To Train and Test Each Identifier.

Every single dataset will be created for both web-page specificity and website specificity. Each dataset that has a web-page level of specificity will be exactly the same as each corresponding dataset at a website level of specificity except it will derive its represented data from the user's website data array instead of the user's web-page data array and it will use the website grand library instead of the web-page grand library. Therefore when explaining how each dataset is created it can be assumed that a dataset will be created for both levels of specificity and whenever a website is referred to it can be that it ambiguously refers to both website and web-page specificity.

4.3.2.1 Creating Datasets to Represent the Simplest Representations of the Browsing History.

First a training set is made by creating an Instance for each user where the Instance has one attribute for every website in the grand library with each attribute value is set to False and the final attribute being the class attribute set to the corresponding user name. Next the first 80% of each user's data array is looped through changing any attribute value in the corresponding Instance to True if the website is found in the user's data array. This creates an instance for each user to represent whether the user visited each website or not. The remaining 20% is then used to create a one test Instance for each user in the same way.

Next a training set is made the same way as before except now every attribute value is set to 0. Then the first 80% of each user's data array is looped through and incrementing the attribute value corresponds to each website in the array. This creates an instance for each user containing exactly how many times each website has been visited by that user. The remaining 20% of each user's data array is then used to create one test Instance for each user in the same way.

4.3.3 Converting the Browser Histories to Represent Sequencing

Datasets will now be created to represent every combination of N-Grams and Skips from a bi-gram 0-skip to a 40-gram 50-skip creating 1900 different training sets.

This is done by creating a loop from 0 to 50 for the skips and then nesting a loop from 2 to 40 for each N-Gram.

Within this nested loop each dataset was created by passing it through an algorithm which had access to each users' data array and the grand library. The algorithm then required two inputs: the number of grams (2-40); the number of skips (0 to 50). The algorithm then created a training set by looping through the first 80% of each users' data array to create an instance where the number of attributes is equal to number of grams with each attribute value containing the integer representation of the website visited in that gram of the sequence. Again the last attribute was the class attribute containing the users name. Then the remaining 20% of each users' data array would be used to create a test set which would contain only instances for each individual user.

To better explain this manipulation lets use this sample data array as a reference :[1,2,3,4,5,6,7]. This sample data array represents a browsing history where the user visited the first seven websites in the grand library. If the algorithm was passed the inputs (2, 1) it would then produce a bi-gram 1-skip dataset. The instances created for this sample data array would be:

1, 3, sampleUser

2, 4, sampleUser

3, 5, sampleUser

4, 6, sampleUser

5, 7, sampleUser

4.4 Format Data So the Classifiers Can Read It

Each classifier that this research uses is conveniently contained within the Weka library. To train and test each classifier Weka requires that the dataset be in special file format called Attribute-Relation File Format (.arff). To format each data set into .arff format I have written a script that takes a dataset as input and writes an appropriate .arff file for it. Each dataset is passed through this script and the .arff files are named according to the dataset that produced it and saved into one directory. This is done for both training and testing datasets.

4.5 Training each Classifier for each Dataset

To train each classifier on every single dataset, I have written a script that will traverse the directory where the .arff files are saved and group each training file with its 9 corresponding testing files. Once grouped it then loops through each training file and passes it

into functions provided by the Weka library that take the training .arff file as an input and creates a model as an output.

4.6 Testing the Identifier

To test each identifier each classifier model that was just created is now tested on each corresponding test file. Each test file contains instances for only one user. A prediction is made for each instance's class attribute (the user name). Whichever user name is predicted the most is considered as the identified user. Since we know exactly which user this is supposed to be record whether the user was correctly identified or not. Once each test file has been tested we write to a results text file both the confusion matrix and the overall accuracy of each identifier. This is then repeated for every training file on every classifier.

4.7 Analyzing the Data

Once all of the results have been calculated and saved another script is used to parse this results file to create a clearer analysis of how each identifier performed. This script has the capability to sort every identifier's results by its overall accuracy. It also can create graphs to show how the increase of N-grams or Skips effects the identifier's accuracy.

5. RESULTS

The following results reflect the amount of users that the identifier correctly predicted for the following classifiers on both levels of specificity.

ZeroR Classifier:
Web-page Specificity:
Guessed 0 right 143 times
Guessed 1 right 1846 times
Website Specificity:
Guessed 0 right 143 times
Guessed 1 right 1846 times

Naïve Bayes Classifier:
Web-page Specificity:
Guessed 1 right 78 times
Guessed 2 right 84 times
Guessed 3 right 84 times
Guessed 4 right 168 times
Guessed 5 right 297 times
Guessed 6 right 1087 times
Guessed 7 right 119 times
Guessed 8 right 72 times
Website Specificity:
Guessed 1 right 3 times
Guessed 2 right 1800 times
Guessed 3 right 142 times
Guessed 4 right 28 times
Guessed 5 right 14 times
Guessed 6 right 2 times

Random Tree Classifier:
Web-page Specificity:
Guessed 1 right 1 times
Guessed 2 right 34 times
Guessed 3 right 230 times
Guessed 4 right 556 times
Guessed 5 right 576 times
Guessed 6 right 394 times
Guessed 7 right 184 times
Guessed 8 right 14 times
Website Specificity:
Guessed 1 right 596 times
Guessed 2 right 1086 times
Guessed 3 right 217 times
Guessed 4 right 73 times
Guessed 5 right 16 times
Guessed 6 right 1 times

Random Forest Classifier:
Web-page Specificity:
Guessed 2 right 2 times
Guessed 3 right 53 times
Guessed 4 right 185 times
Guessed 5 right 588 times
Guessed 6 right 892 times
Guessed 7 right 227 times
Guessed 8 right 42 times
Website Specificity:
Guessed 1 right 749 times
Guessed 2 right 1011 times
Guessed 3 right 147 times
Guessed 4 right 67 times
Guessed 5 right 15 times

K-nearest Neighbors Classifier:
Web-Page Specificity:
Guessed 3 right 20 times
Guessed 4 right 155 times
Guessed 5 right 344 times
Guessed 6 right 590 times
Guessed 7 right 686 times
Guessed 8 right 109 times
Guessed 9 right 85 times
Website Specificity:
Guessed 1 right 125 times
Guessed 2 right 1456 times
Guessed 3 right 249 times
Guessed 4 right 126 times
Guessed 5 right 32 times

Note: Any information omitted can be assumed to have had 0 occurrences.

100% Accuracy was achieved with the K-nearest Neighbors Classifier on the web-page level of specificity for the following datasets:

15, 16 Gram	23-Skip
13-Gram	27-Skip
12, 13 Gram	30-Skip
11-16, 18-19 Gram	31-Skip
13, 16-Gram	32-Skip
16-Gram	33-Skip
10-17 Gram	34-Skip
15-17 Gram	35-Skip
11-16 Gram	36-Skip
14, 15 Gram	37-Skip
13-15-Gram	38-Skip
13, 14-Gram	39-Skip
11-15 Gram	40-Skip
11-14 Gram	41-Skip
12-16 Gram	42-Skip
12-15 Gram	43-Skip

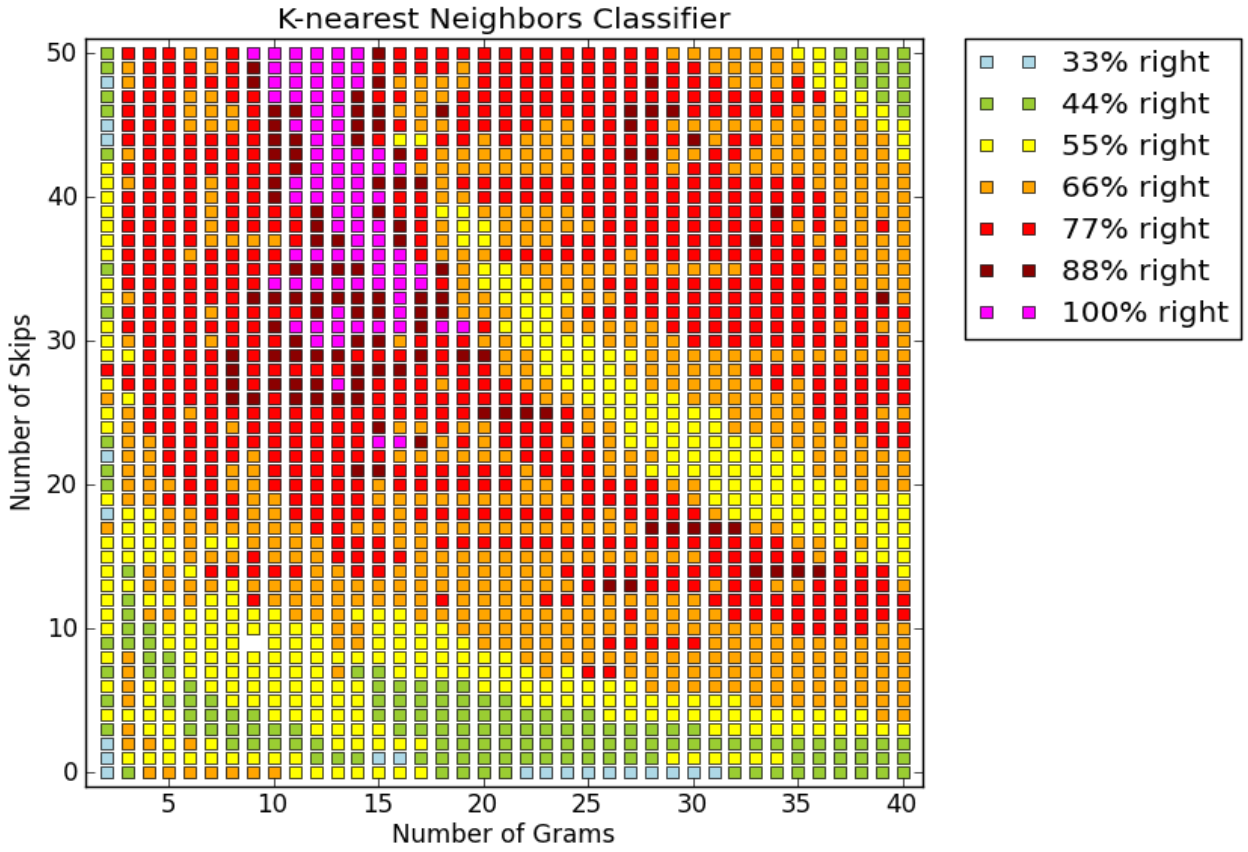

Figure 3. K-Nearest Neighbors Web-page Specificity

12, 13 Gram	44-Skip
11-13 Gram	45-Skip
12, 13Gram	46-Skip
10-13 Gram	47-Skip
10-14 Gram	48-Skip
10-14 Gram	49-Skip
9-14 Gram	50-Skip

Of the K-nearest Neighbor's 85 different datasets that yielded 100% accuracy the 10 with the highest average correct instance prediction are:

13-Gram 49-Skip with an average of 68.322%

16-Gram 36-Skip with an average of 68.335%

15-Gram 42-Skip with an average of 68.352%

17-Gram 34-Skip with an average of 68.36%

15-Gram 43-Skip with an average of 68.384%

17-Gram 35-Skip with an average of 68.389%

14-Gram 48-Skip with an average of 68.424%

14-Gram 50-Skip with an average of 68.432%

16-Gram 42-Skip with an average of 68.441%

14-Gram 49-Skip with an average of 68.467%

6. ANALYSIS/DISCUSSION

Although the results of the experiment are not entirely conclusive certain relationships can be inferred. The data clearly shows that Web-page level specificity in all cases yields a much higher identification accuracy. Initial assumptions persuaded me to believe that website level specificity would have been important, but after reviewing the results it is clear that this is not the case. A possible conclusion to why this is the case is that web-page specificity represents a browsing history that is more unique to the specific user.

After executing the experiment I found that the identifier utilizing the K-nearest Neighbor classifier had the highest overall identification accuracy as well as being the only identifier to achieve 100% accuracy. This suggests that of the tested classifiers the K-nearest Neighbor is the best suited for the task of identification.

There is a certain level of disbelief however associated with the K-nearest Neighbor's accuracy. This disbelief comes from the relatively small amount of collected data that it was tested on. It is not entirely clear if the results observed would be the same if used on a more generalized population. To help minimize this disbelief I have conducted a further analysis of the K-nearest Neighbor Identifier by creating more datasets to test it on. For each of the 85 datasets that yielded 100% accuracy I created 5 new datasets where the 20% of each users history used to test the identifier comes from a different section of the user's browsing history (i.e. the first 20%, the second 20%, etc.) then retested each

test set on each new training set (which is comprised of the other 80% of each users history, to ensure that the datasets being tested were not already used for training). 66 of the 85 original 100% accuracy datasets correctly predicted an average of 90% or higher with the worst accuracy not falling below 83%. Even though no dataset was able to produce 100% accuracy throughout, these results still suggest that the levels of identification accuracy were not just specific to the training sets provided but could be generalized to a larger population of users.

The following Gram and Skip Combinations produced an average identification accuracy of 95% after being retested on the K-nearest Neighbor identifier:

10 Gram 48, 50 Skip

11 Gram 40, 41, 45, 47, 48 Skip

12 Gram 30, 31, 34, 40-43, 45 Skip

13 Gram 27, 30, 31, 34, 42, 45 Skip

14 Gram 31 Skip

Although there is no conclusive results for the exact range of Grams and Skips required to produce 100% accuracy the results from the K-nearest Neighbor suggest that 10 – 14 Grams and somewhere between 31-48 Skips will generally produce the highest identification accuracy.

7. FUTURE WORK

I feel that this research provides a perfect stepping stone for a great deal of future work. If it is proven that a user can be identified by their browsing history alone then there are countless applications where this could be useful. In e-commerce this research could be extended beyond identifying a specific user to the identification of the broader category of user type. If it is then possible to accurately categorize a user based on browsing history you could then begin to see recommendation systems being used across web that would require no input from the user itself. I believe though that the first place that you will see this research implemented will be in advertisements. Ad agencies have already begun to collect and store browsing data from online users and this research would allow the agencies to begin to create user profiles based off of browsing histories. These profiles could then be used to display personalized advertisements.

Another place you could see this research used would be in smarter dating websites. Imagine a dating website where the members could not lie about their personalities because a personality type was determined for them based on their browsing history. This could lead to more honest compatibility tests that link members together based on there intrinsic qualities.

Beyond just e-commerce future work could be done in the field of digital forensics. If each users browsing history literally became a digital fingerprint it would become harder and harder to conceal your identity when you are committing online crimes.

8. CONCLUSIONS

Unfortunately this research did not find an ideal number or range of Grams and Skips that would produce 100% identification accuracy. However the research does answer the question of specificity and provides a strong argument for the use of classifier. However with that being said there are certain fundamental problems with research that could have potentially skewed the results collected.

First the amount of initial data collected is not a sufficient enough to truly say that any of these results could be generalized for all users browsing histories. Due to the insufficient data along with the brute-force nature of this research the recorded results may only be correlated the specific browsing histories of these specific users. Continued research must be done on a larger sample size of users in order to formulate more robust results.

Also this research did not focus on finding the best classifier for the task but rather tried 5 different classifiers on every dataset and chose the one which produced the best results. Again this classifier may only produce these results on this specific data and there is also still the possibility that there is another classifier that would have worked better but just wasn't tested.

Overall this research should be viewed as a pilot study that must be expanded upon before being used in a real life application.

9. REFERENCES

[1] Cavnar, W. B. and Trenkle, J. M. 1994. N-gram- based text categorization. In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval (Las Vegas, NV, 1994), 161–175.

[2] Conti, Gregory. "Googling Considered Harmful." *ACM* (2006). Web.

[3] D. Jang, R. Jhala, S. Lerner, and H. Shacham. An Empirical Study of Privacy-Violating Information Flows in JavaScript Web Applications. In CCS 2010, 2010.

[4] Fayyad, Usama, Gregory Piatetsky-Shapiro, and Padhraic Smyth. "From Data Mining to Knowledge Discovery in Databases." *American Association for Artificial Intelligence* (1996). Print.

[5] Han, J. *Data Mining Concepts and Techniques.* San Francisco: Mourgan Kaufmann, 2001. Print.

[6] Kazunari, Sugiyama, Hatano Kenji, and Yoshikawa Masatoshi. "Adaptive Web Search Based on User Profile Constructed without Any Effort from Users." *ACM* (2004). Web.

[7] M. Jakobsson and S. Stamm. Invasive browser sniffing and countermeasures. Manuscript, 2005

[8] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten (2009); The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1.

[9] *ODP - Open Directory Project*. Web. 22 Nov. 2011. <http://www.dmoz.org/>.

[10] P. Eckersley, “How Unique Is Your Web Browser?,” in Privacy Enhancing Technologies (PET), 2010.

[11] Rastegari, Hamid, and Siti Shamsuddin. "Web Search Personalization Based on Browsing History by Artificial Immune System." *Icsrs* (2010). Web.

[12] Sebastiani, Fabrizio. "Machine Learning in Automated Text Categorization." *ACM Computing Surveys* 34.1 (2002). Web.

[13] Tan, Pang-Ning, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining*. Boston: Pearson Addison Wesley, 2006. Print.

[14] Vincent Toubiana, Arvind Narayanan, Dan Boneh, Helen Nissenbaum, and Solon Barocas. Adnostic: Privacy preserving targeted advertising. In 17th Annual Network & Distributed System Security Symposium, San Diego, CA, USA, March 2010.