A System to Determine Demographic Attributes using Local Browsing History

Siddhesh Karekar
Student
K. J. Somaiya College of Engg.,
Mumbai, India

Amogh Bhabal
Student
K. J. Somaiya College of Engg.,
Mumbai, India

Bhushan Pathak
Student
K. J. Somaiya College of Engg.,
Mumbai, India

Swati Mali
Assistant Professor
K. J. Somaiya College of Engg.,
Mumbai, India

ABSTRACT
The internet is a large storehouse of information. To deliver information efficiently, the audience can be segregated by demographic attributes which can be individually targeted. Companies may be able to obtain or collect information about users' browsing history. Proposed in this paper is a system using TF-IDF and a Neural network, to estimate a user's age, gender and interests by analyzing their browser history.

General Terms
Data mining, neural network.

Keywords
Demographic, browser, history, user, gender, age, estimation, location, prediction.

1. INTRODUCTION
A large amount of information is available to users of the internet. The problem here is that, many times, users may find it difficult to get the information that they are interested in, out of the large pool of information of the internet.

Hence, it becomes important and efficient for companies to provide the user content that is relevant to them. Companies such as advertisement companies might be able to collect information about user browsing history through external sources, or by monitoring user activity across various websites where the ads are displayed. They would want to advertise or provide specific services to interested users.

This can be done by understanding the demographic attributes of the users, such as age, gender and interests.

2. RELATED WORK
The data that will be used to determine demographics mostly contains user browser history. Miha Grčar, Dunja Mladenič, Marko Grobelnik [1] designed a system that will track user browsing history to find user interests. It was implemented as a browser add-on for Internet Explorer.

A. Hawalah & M. Fasli, [2] and C. P. Swami, P. B. Tarte, S. K. Rakshe, S. M. Raut, & N. F. Shaikh [3] have proposed or designed systems that require browser history as well as the timestamp of the webpages visited by user and the duration of the visit.

Generally, the data extracted from the user has to be cleaned, and then used in some preprocessing operations before determining the user interests and demographics. The system designed by Swami et. al. [3] performs filtering of URLs, to remove search engines or sites which are used by all age groups. However, the systems designed by Grčar et. al [1] and Hawalah et. al. [2] perform more complex operations on extracted data, such as the Porter stemming algorithm [4], calculation of TF*IDF, and the creation of ontologies by the former, with the latter implementing K-means on data.

C. P. Swami, P. B. Tarte, S. K. Rakshe, S. M. Raut, & N. F. Shaikh, [5] suggested and compared various algorithms that can be used to determine user age, namely Artificial feedforward neural networks, Apriori algorithm, K-Means Algorithm and ID3 algorithm. A more recent paper by Swami et. al [3] designed a system that determines the user age by extracting user browsing data, and applying Apriori algorithm on the gathered data.

3. APPROACH
The system has been divided into three 'Phases', which flow almost linearly:

Phase 1, 'Extraction': Extraction of browsing data from locally stored browsers, such as Google Chrome;

Phase 2, 'Analysis': Data collection part; downloading and cleaning web pages, creation of reference ontology, and mapping of webpages to ontology; and

Phase 3, 'Prediction': Using mapping of web pages to ontology, determining demographics of user.

3.1 Phase 1 – Extraction
The system detects if a supported browser is installed on the computer, then locates browsing history if available. It can then extract the browser history, wherein it must obtain information about which web page was visited (URL), and how many times it was visited. Additional information such as timestamp of visit, web page title, the number of times the page has been typed into the address bar can also be collected for future use. The user's public IP is also recorded, for IP-based location estimation.

Additionally, if this data is to be used for training purpose, the system requires the user to specify their age and gender before the extraction process begins.

3.2 Phase 2 – Analysis
The steps in the Analysis phase all revolve around two ontologies, the reference ontology and the user ontology.

The reference ontology is a hierarchy of concepts. Concepts can be topics such as 'cooking', 'sports', 'cricket', or objects such as 'bat' or 'ball'. The hierarchy has a tree-like structure; for example, 'sports' could be a parent concept, while 'cricket' and 'football' could be its child concepts, and 'bat' and 'ball' could be the child concepts of 'cricket'. For every concept in the ontology, there is a document containing a number of keywords, or 'terms'. [2] These play a role in the similarity mapping process, described ahead, to identify the interests of user based on the web pages that the user has visited. The

reference ontology needs to be created only once, and can be reused.

The user ontology is created by the same mapping process, of the web pages visited by the user to the reference ontology, which will be described in the following subsections.

3.2.1 Downloading web page contents

The first task in preparing both ontologies involves downloading web page contents. Grčar et. al. [1] and Hawalah et. al [2] used DMOZ, also known as the Open Directory Project. It provides the hierarchy of concepts and also a number of web pages relevant to each concept. The proposed system uses subset of it, relevant to a single application – Shopping, to build a reference ontology.

The system must first download the textual contents of each of those web pages. Then, 'useless' words or stop words, such as 'a' and 'the' must be removed. Finally, the Porter stemming algorithm [4] is applied on the words that remain, to convert words with morphological and inflexional endings to their root word, since they are similar in meaning. For example, words like 'cooking' and 'cooked' will both be converted to 'cook'. This page cleaning process is detailed in the system proposed by A. Hawalah et. al. [2]. The words that remain, called 'terms', are combined to form the document for that concept.

A document is also prepared for each of the web pages in the user's browsing history. This document simply contains all the terms obtained by performing the above cleaning process on its associated web page.

3.2.2 Calculation of Weight for each Term

For every term in all the documents prepared (both at user side and for the reference ontology), a weight is computed. It denotes the importance of the term in that document, and may be different for terms located in different documents. The weight equals the product of Term Frequency (TF) and Inverse Document Frequency (IDF).

TF is calculated by dividing the number of occurrences of the term by the total number of terms in the document.

$$TF = \frac{t_i}{|T_i|} \qquad (1)$$

Where,

t_i = number of occurrences of term t in document i.

$|T_i|$ = Total number of terms in document i.

IDF measures the general importance of a term and is calculated by dividing the total number of documents by the total number of documents that contain the term.

$$IDF = \log\left(\frac{|D|}{|t_j . d_i \in D|}\right) \qquad (2)$$

Where,

$|D|$ = Total number of documents

$|t_{i \cdot d_i} \in D|$ = Number of documents d containing term t.

3.2.3 Similarity Mapping process using Cosine Similarity rule

The next step is to map every document of the reference ontology with every document at the user side. The cosine similarity rule is used for this purpose, as suggested by A. Hawalah et. al [2].

Two documents d1 and d2 can be represented by two vectors containing a number of dimensions equal to the number of combined terms in them. At each dimension, the value of the vector equals weight of the associated term (0, if that term is absent). Then, the similarity is the cosine of the angle between the documents.

$$sim_{cosine}(d_1, d_2) = \frac{\sum_{i=1}^{n} w_{i1} w_{i2}}{\sqrt{\sum_{i=1}^{n} w_{i1}^2} \sqrt{\sum_{i=1}^{n} w_{i2}^2}} \qquad (3)$$

Where,

w_{i1} = weight of ith term of document d1

w_{i2} = weight of ith term of document d2

Thus, the cosine similarity measure of each URL obtained from user with each concept of the reference ontology has been calculated. This value lies in [0, 1], and denotes the degree to which the URL is mapped, or is 'similar' to that concept.

However, this approach assumes that each web page in the history is visited only once, or that the history contains a separate entry for every time the URL is visited. Browsers such as Google Chrome, instead store the number of times a web page is visited as its visit_count.

The proposed system factors the visit count into the similarity as follows:

$$sim = sim * (1 + \ln(visit\_count)) \qquad (4)$$

The log function is chosen to avoid inflation of the similarity for pages visited an extremely large number of times. The base of the logarithm can be chosen depending on how much the page needs to be 'strengthened'. The natural logarithm was found to provide an adequate amount of extra similarity.

3.2.4 Refinement of Mapping Process

Further refinement of the mapping process may be required at this point. A. Hawalah et. al. [2] recommended two methods for doing so; the Gradual Extra Weight Algorithm (GEW) and the Contextual Concept Clustering Algorithm (3C). The 3C algorithm requires detailed information about browsing sessions which may be difficult to obtain. However, the GEW algorithm can be implemented with the available information.

The GEW algorithm assumes that, if a user is interested in a particular concept, then they may be interested in its parent concept. The result values of cosine similarity mapping also called as 'similarity weights', have values within the set [0, 1]. For each parent concept, a portion of the similarity weight of its child concepts will be passed, that is, added to its own similarity weight. This percentage, called as Extra Percentage (EP), is calculated as:

$$EP = \frac{c_i * \alpha}{O} \qquad (5)$$

Where,

c_i = Current level of child concept

O = Total number of levels in reference ontology

The value of Extra Percentage will be different at different levels. It can also be different for child concepts having same parent (thereby being in the same level), if the values of α for said child concepts are given different values.

The parameter α controls how much weight of the child concept to add to the parent concept. It can have values in [0, 1]. A value of 1 transfers the entire weight of child concept to the parent concept. A value of 0 adds no weight from the child

at all. We set it to be 0.5, as experimentally determined by A. Hawallah et. al. to be optimal. [2] After calculating EP, the similarity weight is updated as follows:

$$sim = sim + sim * EP \quad (6)$$

3.3 Phase 3 – Prediction

3.3.1 Determining Demographics

While the process of using a TF-IDF classifier to map similarities to concepts in the reference ontology has been extensively documented, the problem of mapping these results to demographic attributes still remains.

The use of a Multilayer Artificial Neural Network with Resilient Backpropagation was proposed, designed with the following layers:

3.3.1.1 Input Layer

As the first layer of the network, the linear activation function is used and the input is left unchanged. The input layer contains a node for each second-level concept in the reference hierarchy. (The top-level consists of a single root node, Shopping.) Each of its nodes corresponds to each child node of 'Shopping', such as 'Sports', 'Music' and 'Crafts', 35 in all. Each of these nodes must be mapped to a single numeric value.

By the end of phase 2, each concept in the reference ontology is mapped to a number of web pages visited by the user, a similarity value. The GEW algorithm [2] has helped to factor in the weight of the child nodes into the parents, and so only the parent concepts, the 35 children of shopping, have been considered.

However, the input to the neural network must still be a single number. For each node, the mean of the similarities of the concept with all the pages is considered, which then becomes the input to that node.

3.3.1.2 Hidden Layer

The input and output layers have a fixed number of nodes, 35 and 6. To design the hidden layer structure of the neural network, existing architectures such as Jang, Sun, and Mituzani's proposed system for color recipe prediction, which used a 16×18×21×10 neural network [6] were observed.

Tests for different possible hidden layer structures were conducted. The computation time was negligible (<1 millisecond) for each of the structures and was hence not considered as a factor.

Table 1. Comparison of Layer Structures in Training

Hidden Layer Structure	Number of training epochs (Error = 0.01)
10	94
15	76
17	102
20	74
27	77
35	82
10×10	252
20×20	73
20×20×20	88

The hidden layer structure of a single hidden layer with 20 nodes (74 training epochs) was chosen for the neural network. The activation function chosen was the sigmoidal function.

3.3.1.3 Output Layer

The output layer consists of six output nodes, for each of the six possible categories to classify people into - for both males and females, the age groups

- Less than 30 years old,
- Between 30 and 50 years old,
- Greater than 50 years old

The first category was chosen to be relatively larger, assuming that young teens or children do not shop online by themselves.

3.3.2 Training

For training, user data is required. A helper program was already created for this purpose, which allows users to painlessly submit their data while only have to specify their age and gender. The quality of the training data is important, and each of the six groups must be covered as well as possible.

An error rate must be specified during training. A rate of 0.01 was found to be sufficient.

3.3.3 Output Representation

The neural network helps classify the input into each of the six classes. The output rule with the largest firing strength can be chosen as the correctly predicted class. This helps determine the age and gender.

The user interests can be determined by looking at the similarity mapping. Categories which may be more interesting to the user are denoted by greater values. This allows one to target that user with content relevant to those categories.

For the purpose of representation, a tag cloud is used. This tag cloud is populated with values from only the top-level categories, and makes for an intuitive visual aid in understanding user interests.

As discussed earlier, the system also includes a module for IP-based location estimation which may help shed additional light on user characteristics.

4. EVALUATION AND TESTING

The system includes inbuilt logging functions to record the results of each phase. An example case was run through the system to record its working and performance. The test subject was a 21-year old male, with an Indian public IP address.

The following results were recorded on a machine with the following specifications:

- 4 GB RAM
- Intel i3-2120 CPU @ 3.30 GHz
- Windows 10 64-Bit
- 4 Mbps cable internet connection

The system was built entirely in Java and used Oracle 11g XE as a database. It was built as a desktop application for ease and flexibility of implementation, as compared to a browser extension, like the one created by Grčar et. al. [1], which can put constraints upon the implementation.

4.1 Phase 1

Table 2. Phase 1 Operations

Operation	Time Taken
History Export	3.942 seconds
Public IP Export	0.645 seconds
Unify user data	0.230 seconds
Total (including additional overhead)	5.343 seconds

4.2 Phase 2

This is where the heavy chunk of the operations takes place.

4.2.1 Reference Ontology

This process needs to be performed only once.

Table 3. Phase 2 Operations on Reference Ontology

Operation	Time Taken
Download web pages in reference ontology, TF calculation	1 hour 11 minutes 54 seconds
IDF calculation	7 minutes 39 seconds
Weight calculation	19 seconds

The downloading of web pages represents the most erratic component of this process, since at least some websites are bound to have problems, and some links may just be dead.

Individual statistics for the time taken for each web page to be downloaded, and the reasons for being unable to were recorded.

The following chart visualizes the time spent in downloading web pages (in milliseconds), sorted in the order of time taken.

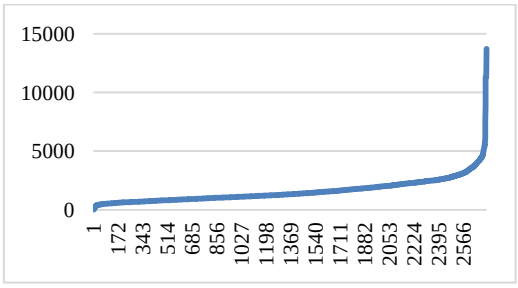

Fig 1: Time taken to download reference web pages

Thus, most web pages took approximately 2 seconds to download, but on the higher side, web pages can take as many as ~14 seconds.

4.2.2 User Ontology

Table 4. Phase 2 Operations on User Ontology

Operation	Time Taken
Download web pages in user ontology, TF calculation	4 minutes 19 seconds
IDF calculation	1 minute 57 seconds
Weight calculation	2 seconds
Similarity mapping	4 hours 29 minutes 1 second
Gradual Extra Weight	4 seconds

The user history included 167 URLs. A similar chart was also constructed for the user ontology, but this can vary greatly in each user's case.

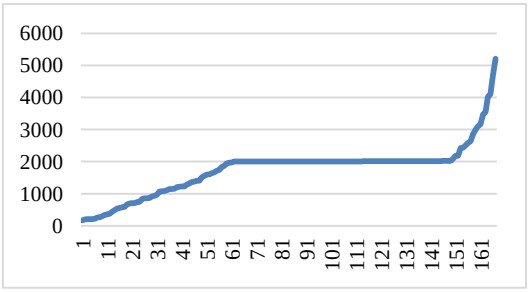

Fig 2: Time taken to download user web pages

The web pages in the user history seemed to be more consistent, with an average time of ~2 seconds.

4.3 Phase 3

Training was performed using a set of 9 training examples. The times taken for each individual operation are as follows:

Table 5. Phase 3 Operations

Operation	Time Taken
Training	91 milliseconds
Interests prediction	4 seconds
Age and gender prediction	1 millisecond
IP-based location estimation	46 milliseconds

The input was correctly classified as the age group 'Young' and the gender 'Male'. The IP-based location estimation module also correctly detected the user to be from India.

5. ISSUES AND CHALLENGES

5.1 Scalability

The biggest issue with the system is scalability. While the system can perform reasonably well to classify in a smaller subject context, such as the chosen case of shopping, a larger context necessitates a larger reference ontology, which means more concepts. This heavily increases the computations (and hence, time) required, especially in similarity mapping, unlike the other tasks involving the reference ontology which happen only once.

The number of URLs in the user history also plays an important role. One must note that every web page visited by the user is mapped with every concept in the reference ontology – this can create an extremely large number of mappings if the user history contains too many URLs. (The number of concepts in the reference ontology, however, remains fixed.)

One way to deal with this is filtering to remove potentially useless URLs before Phase 2 itself. Another way can be combining multiple duplicate URLs and updating their visit counts (as simply as taking the sum of the number of merged URLs).

5.2 Reference Ontology Design

The system contains several constants to tweak the computations in the process, which can greatly hamper the performance if chosen incorrectly.

One major implication of this is in the parameters given to the GEW algorithm. Since the children of the root node in the reference ontology are used as input nodes, a 'wide' ontology will mean more nodes in the input layer.

Conversely, a 'deep' ontology means that the contribution to similarity by the children depends even more heavily on the GEW algorithm's parameters, which may not be desirable.

6. CONCLUSION

There is a large amount of information available on the Internet. Users from different demographic groups have different tastes, and segregating them can be useful to target them separately, and more effectively.

The proposed system has thus demonstrated the use of a TF-IDF and a Multilayer Resilient Backpropagation Neural Network for doing the same, on the basis of web pages visited by them.

The future scope of this system would involve techniques to improve the speed and accuracy of the operations being performed. To enhance speed, implementation of distribution and parallelization can be looked into. To increase the accuracy of the mapping process, further research can be conducted into the values of the parameters involved.

7. ACKNOWLEDGMENT

We would like to show our gratitude to Ms. Smita Sankhe, Assistant Professor, K. J. Somaiya College of Engineering, and Mr. Prasanna Shete, Associate Professor, K. J. Somaiya College of Engineering, for helping us get through a few difficult spots during implementation.

8. REFERENCES

[1] Grčar, M., Mladenič, D., & Grobelnik, M. (2005). User profiling for interest-focused browsing history. In Proceedings of the Workshop on End User Aspects of the Semantic Web (pp. 99-109).

[2] Hawalah, A., & Fasli, M. (2015). Dynamic user profiles for web personalisation. Expert Systems with Applications, 42(5), 2547-2569.

[3] Swami, C. P., Tarte, P. B., Rakshe, S. K., Raut, S. M., & Shaikh, N. F. (2015). Detecting the age of a person through web browsing patterns. International Journal of Computer Applications, 118(12).

[4] Porter, M. (2008). The Porter stemming algorithm, 2005. http://www. tartarus. org/martin/PorterStemmer/index. html. Last visited on July, 16.

[5] Swami, C., Tarte, P., Rakshe, S., Raut, S., & Shaikh, N. F. Detecting the age of a person through web browsing patterns: A Review. International Journal of Soft Computing and Engineering (IJSCE) ISSN, 2231-2307.

[6] Jang, J. S. R., Sun, C. T., & Mizutani, E. (1997). Neuro-fuzzy and soft computing, a computational approach to learning and machine intelligence.