Automatic Web Page Categorization Using Machine Learning and Educational-Based Corpus

Patrick Dave P. Woogue, Gabriel Andrew A. Pineda, and Christian V. Maderazo

***Abstract*—The Internet is a powerful instrument that contains hundreds to thousands of resources. There is a need to categorize these resources based on certain categories in order to organize the contents of the Web better. This research aims to build a corpus that would be representative of pre-defined educational categories. This study will experiment on seven different algorithms that will be able to categorize web pages based on educational domain. Many studies about web categorization have already been conducted but is based on a general set of categories. This research will focus primarily on a predefined set of categories that are closely related to educational domains. With the use of machine learning, the classifier will be able to analyze what a web page is all about and determine its category. The study will also compare the different classifiers used. As a result, the system will be able to assign a web page to a particular educational domain and can be used by schools to determine the categories of web pages frequently requested by students. Linear SVM was also able to build a lexicon for the different categories. The top words for each category were then determined using this lexicon.**

***Index Terms*—Corpus, decision trees, k-nearest neighbor, linear support vector machine, logistic regression, machine learning, multinomial naïve bayes, multilayer perceptron, natural language processing, web page categorization.**

I. Introduction

There are many activities going on in an educational institution whether it's in a high school, college or university. With the rapid growth of technology, one could expect that most of these institutions would have access to the Internet all the time [1]. The Internet today contains an enormous amount of web pages and it would be very difficult to keep track of all the websites accessed by every student. It would also be very impractical to block access to most of the websites because we will never know when a student would really need it. It would also be problematic to give priority to students using the Internet for entertainment purposes over another student who wants to use it for something educational. A solution might be to get faster Internet connection and in fact a number of countries today have very fast connection to the Internet. Unfortunately, this is not a solution especially for third world countries and it is not an economical solution as well due to waste of bandwidth. That is why it is also crucial to be able to provide the best service to the students. This means to prioritize those students accessing the Internet for educational purposes and giving less priority to those using it for entertainment purposes. It would also be helpful for an institution to know what its Internet is actually used for. A university is typically divided into different departments with their own specific technical domain. It would really be useful for a department if they are able to control their Internet to prioritize web pages that are specific to their technical domain. One possible solution to this problem is to use machine learning methods so that a computer would be able to categorize the web page being requested by a student and determine its priority based on the category [2]. This research will not build a system that can control the Internet access based on categories but instead will solve a sub problem to the solution. This research will focus on building a system that can categorize web pages based on a predefined set of categories. There are many methods for a machine to do web page categorization. Many studies have already shown the different possible methods to do so and it is done through machine learning and natural language processing. One study [3] discussed the different algorithms for categorizing and the different features that can be exploited from a web page. Erikkson's study [4] attempted to categorize web pages through a set of general categories using different text classification algorithms and compared the results. An in depth study on web page classification wherein it demonstrates the methods to preprocess the data, the different classifiers to use and ways for evaluation has also been done [5]. The study will experiment on different machine learning algorithms in order to classify web pages according to educational domains. This research will use 9 predefined set of categories that are closely related to educational domains. These are namely Architecture & Fine Arts / Design, Math and Science, Arts (Psychology, Literature, History), Medical, Politics (Law & Governance), Business and Economics, Nutrition / Diet / Health, Software Engineering / Programming / Technology and Others which means that it doesn't fit a certain category. The classifier used will be multiclass meaning that it can only assign one category per web page. The usefulness of this research will only be fully realized once it is combined with a system that can be programmed to prioritize a set of categories when there is a huge amount of network traffic in an institution. This research aims to build a corpus for each of the categories, and this research also aims to develop a classifier system that categorizes web pages based on educational domain using different machine learning algorithms. The system will also be evaluated on various evaluation methods such as precision and recall.

Manuscript received October 9, 2017; revised December 11, 2017.

Patrick Dave P. Woogue, Gabriel Andrew A. Pineda, Christian V. Maderazo are with University of San Carlos, Philippines (e-mail: patwoogue@gmail.com).

II. Related Review of Literature

There are similar studies that have been done about web page categorization. The studies differed with the technique and methods used. The studies also varied in terms of the accuracy and the use of different machine learning algorithms.

A. Algorithms and Methods for Web Page Categorization

(SVM) classifiers to classify web pages using both their text and context feature sets. The researchers used the WebKB data set to experiment the classifier. The results showed that when compared with the FOIL-PILFS, the SVM performed very well even when using the text components only. It also showed that using context features which consisted of title components and anchor words improved the classification accuracy significantly

Kwon & Lee [6] developed a web page classifier that is based on an adoption of k-Nearest Neighbor (k-NN) approach. The research supplemented the k-NN approach with a feature selection method and a term-weighting scheme using markup tags in order to improve the performance of k-NN approach.

Patil & Pawar [7] performed a study that attempted to classify web pages using the Naïve Bayesian algorithm. The research considered ten categories to be classified and the NB algorithm had an accuracy of 89.05% accuracy. It was also observed that the classification accuracy was proportional to the number of training documents.

There are more similar studies that used a variety of methods for web page categorization. Mahdy & Qader [8] proposed a system to classify web pages using Neural Networks. Shibu, Vishwakarma & Bhargava [9] used a combined approach of Page Rank and Feature Selection. Asirvatham & Ravi [10] proposed a method that made use of other information in a web page such as images, audio and video. Roul & Sahay [11] proposed a system that made use of the frequent item word sets generated by the Frequent Pattern Growth.

B. Feature Selection and Feature Extraction Techniques

This section lists different techniques in choosing the features from a given text in order to give a comparison of the techniques used in this study such as TF-IDF. Riboni [12] tested five different It concluded that the combination of hypertextual and local representation of web pages can improve classification accuracy. They also introduced a new method for representing linked pages using local information that can make hypertext categorization feasible for real-time applications.

Sarode & Gadge [13] conducted a research that described an approach a hybrid approach for dimensionality reduction in web page classification using a rough set of naïve Bayesian method. Dimensionality reduction is important since web pages tend to have a great number of terms and this may cause problems such as processing time. In this study, a Quick Reduct algorithm was used for dimensionality reduction and information gain was used for feature selection. The study concluded that this approach would improve the accuracy and efficiency of the classifier

Rajalakshmi & Aravindan [14] performed a study that used only the URL of web page as the feature. This has a great advantage because the contents of a web page need not be fetched. This paper proposed an approach to web page classification based on features extracted from URLs alone. The results of the study achieved values of 0.7, 0.88 and 0.76 for Precision, Recall and F-measure values respectively.

There are a lot of researches that made use of Feature Selection and Feature Extraction techniques. Rogati & Yang [15] conducted a study on a large number of filter feature selection methods for text classification. Ren & Zhang [16] performed classification through the use of an improved bloom filter algorithm and an improved feature weight algorithm that is based on the characteristics of the web page. Kan & Thi [17] demonstrated the usefulness of the uniform resource locator (URL) alone in performing web page classification.

C. Evaluation of Web Page Classifiers

Costa, Lorena, Carvalho & Freitas [18] reviewed some evaluation metrics used to evaluate hierarchical classification models. The different evaluation metrics used were the Flat Performance Measures, Hierarchical Performance Measures, Distance-based Measures, Depth-dependent Measures, Semantics-based Measures and Hierarchy-based Measures. The observation of the study was that there is not yet a consensus concerning which evaluation measure should be used in the evaluation of a hierarchical classifier.

Yang [19] showed a comparative evaluation of a wide-range of text categorization methods. The results of the study showed that as a global observation kNN, LLSF, and a neural network method had the best performance while the other algorithms performed relatively well except for the Naïve Bayes approach.

Li & Yang [20] presented a formal analysis of popular text classification methods which are Support Vector Machines, Linear Regression, Logistic Regression, Naïve Bayes, K-Nearest Neighbor and Multi-class Prototype classification. The study shows that the performances of regular LLSF, Neural Network, Linear Regression and SVM were close to each other.

III. Methodology and Results

This section discusses the study's methodology on how to train the classifier and to implement and test the system. This chapter enumerates the steps needed to be able to build a classifier. The figures in section B do not show the actual data and values that occurred in the research since it would be too much to show in this paper. But it is enough to give a good idea of what is happening.

Fig. 1 shows the conceptual framework of this study. Harvesting web pages as data is required in order to build the corpus and to train the system. The researchers would manually gather the web pages and categorize it accordingly. After the collection of web pages, a series of procedures need to be done in order to build the corpus. The corpus is essentially the input to the machine algorithm in order to obtain the trained classifier. After the classifier is obtained, it will then be tested using a number of evaluation metrics.

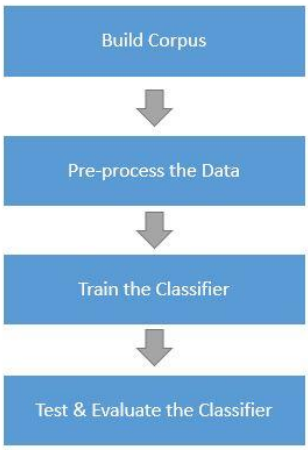

Fig. 1. Conceptual framework.

A. Building a Corpus for the Different Categories

In building the corpus, the researchers manually collected web pages from the Internet based on the different categories and gathered 150 web pages per category. Most of the web pages gathered for every category originated from www.wikipedia.org. Table I shows the different categories, the top 3 websites and the number of web pages gathered.

TABLE I: NUMBER OF WEB PAGES PER CATEGORY

Category	Top 3 websites	# of pages
Architecture, Fine Arts & Design	www.wikipedia.org www.designbuildings.co.uk www.house-design-coffee.com	69 15 13
Computer & Information Science	www.wikipedia.org www.geeksforgeeks.org www.w3schools.com	83 6 5
Politics (Law & Governance)	www.wikipedia.org www.dictionary.law.com www.polisci.duke.edu	57 26 13
Engineering	www.wikipedia.org www.engineersedge.com www.engineeringtoolbox.com	101 25 11
Business & Economics	www.wikipedia.org www.economist.com www.investopedia.com	74 26 11
Arts (Psychology, Anthropology, Philosophy & etc…)	www.wikipedia.com www.allpsych.com www.alleydog.com	54 26 26
Math & Science	www.wikipedia.com www.thoughtco.com www.hach.com	122 27 1
Medical	www.wikipedia.com www.emedicinehealth.com www.aboutmedicalschools.com	120 25 2

B. Gather the Training and Testing Web Pages

1. Retrieve raw contents and extract HTML tags. After building the corpus, the raw contents of all the web pages were extracted using the Python BeautifulSoup library. The raw contents were then further processed by selecting sentences and paragraphs from a specific set of HTML tags while the rest of the content was removed. The HTML tags used were the p, h1, h2, h3, h4 and h5.

http://archidose.blogspot.com/

A Daily Dose of Architecture (Almost) daily architectural musings and imagery from New York City Pages My recent posts at World-Architects Saturday, August 05, 2017 Book Review: Magnetic City Tuesday, August 01, 2017 Two New Books Sunday, July 30, 2017 Finding Fay: Cooper Chapel Saturday, July 29, 2017 Finding Fay:

Fig. 2. Sample website after extracting HTML tags.

2. Split data and load contents and splitting into train and test set. The web pages were represented as integers that would be mapped to a line in a file to get retrieved contents from the previous step. A list of web pages was created together with a list of their corresponding categories. Both list were then split into a training and testing set with a percentage of 70% and 30% respectively. The actual contents of the web pages were then loaded into the list.

X_train =
[131, 72, 41, 133, 9, 59, 126, 66]
y_train =
[1, 2, 3, 4, 5, 6, 7, 8]

X_test =
[106, 77, 2, 123, 83, 120, 85, 97,]
y__test =
[1, 2, 3, 4, 5, 6, 7, 8]

Fig. 3. After splitting the data into training and testing set.

3. Remove stop words and perform lemmatization. After the actual contents were loaded, the stop words of each content were removed. Lemmatization was also performed to convert similar words to its base form using the Python spaCy library.

TABLE II: AFTER REMOVING STOP WORDS AND PERFORMING LEMMATIZATION

Before	Remove stop words	Perform lemmatization
"Architectural design values Architectural design values make up an important part of what influences architects and designers when they make their design decisions. However, architects and designers are not always influenced by the same values and intentions."	“Architectural design values Architectural design values important influences architects designers design decisions . , architects designers influenced values intentions .”	“architectural design value architectural design value important influence architect designer design decision architect designer influence value intention”

4. Convert into bag-of-words with TF-IDF. The list of web pages in the training set was then converted into a bag-of-words. The bag-of-words were combined to form a set of words that would represent each web page. The web pages in the training and testing list was mapped to these set of words. Instead of counting word frequency, the TF-IDF weighting scheme was used to assign weights to each word. The data is now ready to be inputted to train the different classifiers.

{('value', 0.0281), ('architect', 0.0563), ('intention', 0.0732), ('important', 0.1187), ('influence', 0.0672), ('architectural', 0.0563), ('design', 0.0511), ('decision', 0.0314), ('designer', 0.0267)}

Fig. 4. After converting into bag-of-words with TF-IDF.

C. Train the Classifier Using the Input Data

In training the classifier, seven supervised machine

learning algorithms were used namely Logistic Regression, Linear SVM, Multinomial Naive Bayes, k-Nearest Neighbor, Decision Trees, Random Forest and Multilayer Perceptron. The Python library used was Scikit-learn to perform the training of the different classifiers. These classifiers accept different parameters to fine tune the algorithm. This research does not focus on all parameters and only uses the most common ones for each classifier. Table III gives a summary of the parameters that were used. The training set was used to choose the best parameter for each classifier based on a given set of parameters. The training set was further split into a training and validation set. The technique used was the stratified k-fold cross-validation where the training set is split multiple times to choose the best classifier with a given set of parameters. Logistic Regression and Linear SVM were tuned using the same parameter C, which determines the strength of regularization. A high value of C correspond to less regularization. Multinomial Naive Bayes was tuned using the parameter alpha, which controls the model complexity. A large alpha means more smoothing which results in less complex models. K-Nearest Neighbors was tuned using the parameter n_neighbors which specifies the number of neighbors to use. Decision Trees was tuned using the parameters max_depth and max_leaf_nodes which means the maximum depth of the tree and create a tree with that much leaf nodes, respectively. Random Forests was tuned using the parameters n_estimators and max_features which means the number of trees in the forest and the number of features to consider when looking for the best split, respectively. Multilayer Perceptrons was tuned using the parameter hidden_layer_sizes which is a tuple of the form (i1,i2,i3,...,in). This gives a network with n hidden layers, where ik gives you the number of neurons in the kth hidden layer.

TABLE III: SCIKIT-LEARN IMPLEMENTATION

Classifier	Parameter Grid	Best Parameter	Accuracy
Logistic Regression	{'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000]}	{'C': 100}	0.913
Linear SVM	{'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000]}	{'C': 1}	0.912
Multinomial Naive Bayes	{'alpha': [0.001, 0.01, 0.1, 1, 10, 100, 1000]}	{'alpha': 0.1}	0.908
k-Nearest Neighbors	{'n_neighbors': [3, 5, 7, 9]}	{'n_neighbors': 9}	0.869
Multilayer Perceptrons	{'hidden_layer_sizes': [100, (100, 100), (100, 100, 100)]}	{'hidden_layer_sizes': 100}	0.914
Decision Trees	{'max_depth': [None, 2, 5, 10], 'max_leaf_nodes': [None, 5, 10, 20]}	{'max_depth': 10, 'max_leaf_nodes': 20}	0.731
Random Forests	{'n_estimators': [10, 20, 30], 'max_features': ['sqrt', 'log2']}	{'n_estimators': 30, 'max_features': 'sqrt'}	0.820

D. Test and Evaluate the System

After training the classifier, the testing set was fed to the different classifiers. The classifier that had the highest accuracy was Linear SVM with a score of 0.931. Table IV shows a summary of the evaluation of the different classifiers. It shows the accuracy, precision, recall, fscore of all the classifiers and they are divided into two namely micro and macro. Since Linear SVM had the highest score for all the metrics compared to the other classifiers, additional details about the classifier especially the top 15 features learned per category is shown on Table V and its confusion matrix is shown on Table VII. Linear SVM was able to build a Lexicon for the eight categories. The lexicon consists of approximately 9,000 words. A snippet of the lexicon is shown at Table VI. As shown in the table, the word architecture has the highest weight for category 1 compared to other categories which shows its relevance for category 1.

TABLE IV: EVALUATION OF DIFFERENT CLASSIFIERS

Classifier	Accuracy	Precision (micro)	Recall (micro)	Fscore (micro)	Precision (macro)	Recall (macro)	Fscore (macro)
Logistic Regression	0.925	0.925	0.925	0.925	0.926	0.925	0.925
Linear SVM	0.931	0.931	0.931	0.931	0.932	0.931	0.931
Multinomial Naive Bayes	0.917	0.917	0.917	0.917	0.919	0.917	0.917
k-Nearest Neighbors	0.883	0.883	0.883	0.883	0.900	0.883	0.887
Decision Trees	0.758	0.758	0.758	0.758	0.784	0.758	0.763
Random Forests	0.814	0.814	0.814	0.814	0.822	0.814	0.815
Multilayer Perceptrons	0.917	0.917	0.917	0.917	0.917	0.917	0.916

TABLE V: TOP 15 FEATURES FOR LINEAR SVM

(1) Architecture, Fine Arts & Design	(2) Arts (Psychology, Anthropology, Philosophy and etc)	(3) Business & Economics	(4) Computer & Information Science
architecture	philosophy	business	computer
art	anthropology	economic	programming
architect	psychology	management	datum
architectural	social	franchise	software
color	browser	market	robot
sketchup	behavior	cost	information
draw	sociology	commercial	stack
artist	mental	commerce	sign
design	child	shareholder	computing
landscape	linguistic	price	learn
fashion	glossary	accountability	github
building	disorder	collar	network
kit	philosophical	hr	library
landscaping	counseling	company	computational
structural	crowd	key	hardware

(5) Engineering	(6) Math and Science	(7) Medical	(8) Politics (Law & Governance)
engineering	physics	medical	political
engineer	theory	drug	law
chemical	physic	medicine	legal
telephone	list	disease	governance
tool	topic	nursing	international
fastener	chemistry	health	trump
process	mathematic	specialty	ideology
quality	perfect	pharmacy	peace
flow	prime	microscopy	government
signal	axiom	occupational	slate
cad	reciprocity	clinical	global
freeze	ring	patient	relation
unit	acoustic	physician	court
domain	cardinal	biomedical	lawyer
calculator	motion	medterm	fraud

TABLE VI: LINEAR SVM LEXICON

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	Words
-0.02	0.04	-0.05	-0.06	-0.11	0.09	0.14	-0.04	archaeological
0.11	0.18	-0.05	-0.05	-0.02	-0.02	-0.04	-0.02	archaeologist
0.19	0.07	-0.08	-0.04	-0.05	0.05	-0.03	-0.07	archaeology
-0.02	0.20	-0.03	-0.03	0.003	-0.02	-0.03	-0.02	archaic
-0.01	-0.01	-0.02	0.047	0.006	-0.03	-0.01	-0.01	archimede
7.28	-1.39	-1.49	-2.11	-4.33	-1.5	-1.46	-1.73	architect
0.002	-0.03	-0.05	-0.01	-0.03	0.14	-0.01	-0.01	architectura
7.52	-1.79	-1.39	-0.58	-6.40	-1.9	-1.3	-1.53	architectural
15.24	-4.42	-2.47	-3.20	-4.15	-5.1	-2.68	-2.48	architecture
-0.30	0.38	-0.39	-0.38	-0.14	-0.1	-0.37	1.26	archive
-0.46	-0.38	1.68	-0.62	-0.07	-0.4	-0.27	-0.6	area

TABLE VII: CONFUSION MATRIX FOR LINEAR SVM CLASSIFICATION

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
40	2	1	0	2	0	0	0
0	44	0	0	0	0	1	0
0	2	41	0	0	2	0	0
1	0	0	42	0	0	2	0
0	0	0	0	42	2	1	0
0	0	1	0	0	42	2	0
0	0	0	3	1	1	40	0
0	0	0	0	0	0	1	44

IV. CONCLUSION, SUMMARY & FUTURE WORK

Linear models such as Logistic Regression and Linear SVM showed to have good accuracy when used for text classification. This makes sense since linear models can scale to very large datasets and work well with sparse data. Multinomial Naive Bayes also showed to have a good accuracy and they are also very similar to linear models. Naive Bayes models work very with high-dimensional sparse data as well. Multilayer Perceptrons also had a very good accuracy of 0.917. This research did not go into detail with Neural Networks and this can be further improved. Although k-Nearest Neighbors are known to perform badly on sparse datasets, it was able to get a respectable accuracy of 0.883. Decision Trees and Random Forests are at the bottom two in terms of accuracy when compared to other classifiers.

This study was able to build a corpus for the different educational categories. The corpus that was built proved to be a good representation for the different educational categories as shown by the top features that was learned by Linear SVM. The study was able to train 7 different classifiers namely Logistic Regression, Linear SVM, Multinomial Naive Bayes, k-Nearest Neighbors, Random Forests, Decision Trees and Multilayer Perceptrons. The researches were only able to tune few of the parameters for these classifies and can be further improved. Furthermore, the research study concludes that linear models proved to be good models for text classification and is a good supervised machine learning algorithm for web page categorization. According to literature, linear models are known to be very fast to train and also fast to predict.

This research study can be extended by adding more educational categories in order to make it more specific. An example would be to split the category "Math and Science" and treat them as different categories. The study also did not take into account the overlap of the different categories such as Math and Engineering. The building of the corpus can also be further improve extended by gathering more web pages for each categories and to see the effects on the accuracy of the different classifiers. Most of the web pages came from www.wikipedia.org and this can be further improved by collecting information from other websites as well. The preprocessing step can be further improved by using a bag-of words with more than one word. Other weighting schemes aside from TF-IDF can also be explored to see if there is an improvement in accuracy. The study can also be extended by training the data with other classifiers not used in this research. The different classifiers used in this study can be further improved as well by fine tuning the different parameters. Lastly, the research can also be extended to create a system that can filter web pages according to their educational categories. This can greatly help different departments in colleges maximize the efficiency of their bandwidth usage.

ACKNOWLEDGMENTS

We wish to thank the Department of Computer and Information Sciences, University of San Carlos for the research opportunity.

EFERENCES

[1] R. Fleck and T. McQueen. (1999). Internet access, usage and policies in college and universities. [Online]. Available: http://www.firstmonday.org

[2] M. Tsukada, T. Washio, and H. Motoda, "Automatic web-page classification," *Lecture Notes in Computer Science (LNCS)*, 2001.

[3] X. Qi and B. D. Davison, "Web page classification: features and algorithms," *ACM Computing Surveys*, 2009.

[4] T. Eriksson, *Automatic Web Page Categorization Using Text Classification Methods*, KTH Royal Institute of Technology, 2013.

[5] B. C. Yao, "Web page classification," *Foundations and Advances in Data Mining*, 2005, ch. 9.
[6] O. W. Kwon and J. H. Lee, "Web page classification based on k-nearest neighbor approach," *IRAL*, 2003.
[7] A. S. Patil and B. Pawar, "Automated classification of web sites," *IMECS*, 2012.
[8] Q. S. Mahdy and K. Qader, "Web page classification by using neural networks," *Journal of Pure Applied Sciences*, 2011.
[9] S. Shibu, A. Vishwakarma, and N. Bhargava, "A combination approach for web page classification using page rank and feature selection technique," *International Journal of Computer Theory and Engineering*, 2010.
[10] A. P. Asirvatham and K. K. Ravi, *Web Page Categorization Based on Document Structure*, 2002.
[11] R. K. Roul and S. Sahay, "An effective approach for web document classification using the concept of association analysis of data mining," *International Journal of Computer Science Engineering and Technology*, 2014.
[12] D. Riboni, "Feature selection for web page classification," *EURASIA-ICT*, 2002.
[13] S. Sarode and J. Gadge, "Approach for dimensionality reduction in web page classification," *International Journal of Computer Applications*, 2014.
[14] R. Rajalakshmi and C. Aravindan, "Naive bayes approach for website classification," *Information Technology and Mobile Communication*, 2011.
[15] M. Rogati and Y. Yang, "High-performing feature selection for text classification," in *Proc. the Eleventh International Conference on Information and Knowledge Management*, 2002.
[16] D. Shen, Z. Chen, Q. Yang, H. J. Zeng, B. Zhang, Y. Lu, and W. Y. Ma, "Web-page classification through summarization," in *Proc. the 27th Annual International Conference on Research and Development in Information Retrieval (SIGIR'04)*, 2004.
[17] M. Y. Kan and H. O. Thi, "Fast webpage classification using URL features," in *Proc. Conference on Information and Knowledge Management*, 2005.
[18] E. P. Costa, A. C. Lorena, A. Carvalho, and A. A. Freitas, "A review of performance evaluation measures for hierarchical classifiers," in *Proc. Evaluation Methods for Machine Learning II: Papers from the AAAI-2007 Workshop, AAAI Technical Report WS-07-05*, 2007.
[19] Y. Yang, *An Evaluation of Statistical Approaches to Text Categorizaiton*, Kluwer Academic, 2000.
[20] Y. Yang and X. Liu, "A re-examination of text categorization methods," *Special Interest Group on Information Retrieval (ACM SIGIR)*, 1999.

Patrick Dave P. Woogue was born in Cebu City, Philippines in 1997. He is currently a bachelor of science in computer science at the University of San Carlos. He was an trainee at Advanced World Systems (AWS), located at Cebu City, on the summer of 2017. He is currently an intern at Synacy Inc., also located at Cebu City, where he works as a software developer. His research interests include Neural Networks, Artificial Intelligence and Machine Learning.

Gabriel Andrew A. Pineda was born in Cebu City, Philippines in 1998. He is currently a bachelor of science in computer science at the University of San Carlos. He was an trainee at Rococo Global Technologies Corporation, located at Cebu City, where he worked as a software developer on the summer of 2017. His research interests include artificial intelligence, machine learning and natural language processing.

Christian V. Maderazo was born in Cebu City, Philippines in1972. He graduated with a degree in bachelor of science in computer engineering at the University of San Carlos (USC) on 1996. He attained the master's degree in engineering with the course of computer engineering at USC on 2008. He is currently working as a college instructor in USC. His publications include the following: File Management System for P2P Environment using Lossless Compression Algorithm (Proceedings of the National Conference on Information Technology Educator, 2016); An Analysis of Public Perceptions for K12 Implementation in the Philippines using Web Content Mining Techniques and Wrapper Induction Algorithm(Proceedings of the National Conference on Information Technology Educator, 2016).