ISSN 2395-8618

Comparison of Classification Algorithms for File Type Detection A Digital Forensics Perspective

Konstantinos Karampidis, Ergina Kavallieratou, and Giorgos Papadourakis

Abstract— **Digital forensics is a relatively new field in Computer Science and it focuses on the acquisition, preservation and analysis of digital evidence, in a way that that these evidences are suitable for presentation in a court of law. Forensic investigators follow a standard set of procedures. One major and difficult problem is the correct identification of file types. Criminals often hide evidence in a digital device, by changing the file type. It is very common, a child predator to try to hide image files with immoral content in order to fool police authorities. In this paper we examine a methodology for file type identification, which uses computational intelligence techniques for feature selection and classification. This methodology was applied to the three most common image file types (jpg, png and gif). In order to ascertain the method's accuracy, different machine learning classifiers were utilized. A three stage process involving feature extraction (Byte Frequency Distribution), feature selection (genetic algorithm) and classification (decision tree, support vector machine, neural network, logistic regression and k-nearest neighbor) was examined. Experiments were conducted having files altered in a digital forensics perspective and the results are presented. The examined methodology showed -in most cases- very high and exceptional accuracy in file type identification.**

Index Terms— **artificial neural network, computer crime, digital forensics, file type detection, genetic algorithms, machine learning algorithms**

I. Introduction

DIGITAL forensics concerns the recovery and investigation of possible evidence found in digital devices. A digital forensics examination typically consists of four major phases; data collection, examination, analysis and report. Data examination is very critical because in this phase the evaluation of the collected data will be made. Therefore misjudgment of possible evidence may lead a court of law to wrong conclusions about criminal's activities. For example, it is very common, a child predator to try to hide image files with immoral content in order to fool police authorities. Typically they change file's extension, file's signature (magic bytes) or even both. On the other hand, forensic examiners use specialized forensic software to identify those forged files but in some cases even the best forensic software cannot identify correctly a file type. File type detection methods can be classified into three categories: extension, magic and content based methods. A lot of scientific methods have been proposed [1] but although some of them show more advantages than weaknesses, none is comprehensive or reliable enough to fulfil all the requirements. Due to the fact that extension-based methods are easy enough to be spoofed – just by a simple renaming of the file- and since the magic bytes in files cannot precisely determine true file type (because there is no predefined standard for the developers), the most significant methods are those who focus on the content of files. To achieve this, the content of each file is examined thoroughly and statistical modeling techniques are utilized to detect the correct file type. These methods are the most promising ones and proved to show the best results. M. McDaniel et al. [2], [3] suggested three algorithms for content-based file type detection. The correctness varied from 23 % to 96 % depending upon the algorithm used. W.J. Li et al. [4] worked on these algorithms and proposed to compute a set of centroid models and use clustering in order to find a minimal set of centroids with good performance while the use of more pattern data was considered necessary. Their methodology had a result of 82 % to 89.5% accuracy (one or multi centroid respectively) and 93.8 % accuracy when they examined a larger number of files in the dataset. Supervised learning techniques were used by J. Dunham et al.[5]. More specifically they used neural networks for classification and reached 91.3 % accuracy. M.C. Amirani et al. [6] used the Principal Component Analysis and unsupervised neural networks for the automatic feature extraction. They also used a neural network for classification, succeeding an accuracy of 98.33 % which was the best so far. D. Cao et al. [7] used Gram Frequency Distribution and vector space model with results of 90.34 % accuracy. I. Ahmed et al. [8] proposed two very interesting methods. Primary they used the cosine distance as a similarity metric when comparing the file content. Subsequent they decomposed the identification procedure into two steps. They used 2000 files of 10 file types as a dataset and achieved an accuracy of 90.19 %. I. Ahmed et

This work was submitted for review on 8/8/2016.

Konstantinos Karampidis is with Department of Information & Communication Systems Engineering, University of the Aegean, 83200 Karlovasi Samos, Greece (e-mail: karampidis@ outlook.com).

Ergina Kavallieratou is with Department of Information & Communication Systems Engineering, University of the Aegean, 83200 Karlovasi Samos, Greece (e-mail: kavallieratou@aegean.gr).

Giorgos Papadourakis is with Department of Informatics Engineering, Technological Educational Institute of Heraklion, 71500 Heraklion Crete, Greece (e-mail: papadour@cs.teicrete.gr).

ISSN 2395-8618

al. [9] also proposed two new techniques to reduce the classification time. The first method involved a feature selection technique and the K-nearest neighbor (KNN) as a classifier. The second method was the content sampling technique, which used a small portion of a file to obtain its byte-frequency distribution. M.C. Amirani et al.[10] then proposed an improved version of their first method by using a Support Vector Machine classifier and finally succeeded in raising the accuracy of the method to 99.16 %. J. D. Evensen et al. [11] used an n-gram analysis with naïve Bayes classifier to a large dataset of 60000 files (6 file types) with very good results achieving 99.51 % topmost. Finally, we proposed a new method [12] which included a three stage process involving feature extraction (Byte Frequency Distribution), feature selection (genetic algorithm) and classification (neural network). This method was tested to a large dataset in a digital forensics perspective and it showed extremely high accuracy (99.61%). All above papers refer to identification of whole files. Although our method, achieved extremely high accuracy even in a digital forensics perspective, we considered wise to explore whether the utilization of another classification method achieves better results. More specifically, we will reproduce the first two stages of the method and examine five different classification algorithms - decision tree, support vector machine, neural network, logistic regression and k-nearest neighbor- on the same dataset. Eventually an evaluation will be made about which classifier shows the best results for a forensic file type detection. A similar work [13] has been published but the authors examined fragments of files. The number of files they examined were too small (150 files of each type at topmost) in order to estimate accurately the correct file type and they used a different method for feature extraction (PCA). Finally the file types chosen for this study included PDF documents, JPG images, ASCII text files (TXT), Microsoft Word documents (DOC), HTML pages and executable files (EXE) which is far different from our point of interest. The rest of the paper is organized as follows: In Section 2 the proposed methodology is described, in Section 3 the different classifiers and the experimental parameters which utilized are briefly described and in Section 4 the experimental results are presented followed by conclusions.

II. Methodology of the Examined Method

Due to their importance to Digital Forensics, the identification of the most common image file types (jpg,png,gif) will be examined. The scientific method we examine in this paper uses computational intelligence techniques in order to identify the file type and to reveal the correct type if the file is altered. It involves feature extraction, feature selection and classification. Primarily all files from the dataset are loaded and the features are extracted. Byte Frequency Distribution (BFD) is used as feature extraction method. The number of incidences of each byte value in an input file is counted and an array with elements from 0 to 255 is created. Then each element of the array is normalized by dividing with the maximum occurrence. The final result is a file containing 256 features for each instance. Subsequently, feature selection is performed using a genetic algorithm. Genetic algorithms can provide candidate solutions. Each candidate solution (chromosome) is represented by a binary feature vector of dimension 256, where zero (0) indicates that the respective feature is not selected, and one (1) indicates that the feature is selected. The score of each candidate solution is evaluated by a fitness function. As a fitness function the Correlation based Feature Selection (CFS) [14] algorithm is utilized. This algorithm evaluates the candidate solutions from the genetic algorithm and choses those which include features highly associated to the file type category and low correlated with each other, by calculating each candidate's solution merit. Let S be a candidate solution consisting of k features. The merit of each candidate solution is calculated as shown in (1).

$$Merit S_{k} = \frac{k\overline{r_{cf}}}{\sqrt{k+k(k-1)\overline{r_{ff}}}} \qquad (1)$$

,where:

$\overline{r_{cf}}$ is the average value of all feature-classification correlations and

$\overline{r_{ff}}$ is the average value of all feature-feature correlations.

CFS stops when five consecutive fully expanded candidate solutions show no improvement. The utilization of the genetic algorithm as a search method and CFS as an evaluator leads to the reduction of the 256 extracted features to 44. Finally a one hidden layer neural network using the backpropagation algorithm is used for classification. Caltech 101 [15] from Caltech University is utilized as dataset. This dataset contains 9144 images in jpeg format from 101 categories. From these jpeg images 5519 of them are utilized. One third of these 5519 files are converted to png format and a similar number to gif format. The dataset is divided into a training set (70%) and a test set (30%). Furthermore, 1840 pdf files were added. The created dataset is uniformly distributed and its exact numbers are indicated in Table I.

TABLE I
THE DATASET

Dataset			
Total files		**Training**	**Testing**
jpeg	1840	1288	552
png	1840	1288	552
gif	1839	1287	552
pdf	1840	1288	552
Total	**7359**	**5151**	**2208**

In this paper the first two phases of the experiment will be reproduced (feature extraction and feature selection) and the performance of five different machine learning algorithms - including the neural network originally proposed in [12]- will be evaluated.

ISSN 2395-8618

III. Learning Algorithms and parameters setup

Waikato Environment for Knowledge Analysis (Weka) [16], an open source machine learning software developed at the University of Waikato, New Zealand was used for all the conducted experiments. An attribute selected classifier was used in Weka. Furthermore, a genetic algorithm was chosen as a search method. The population size was 256, the number of generations 100, crossover was set to 0.8 and mutation probability to 0.033. CFS was the fitness function, roulette wheel selection was used to probabilistically select individuals and the single-point crossover operator was selected. The use of CFS as a filter selection evaluator and the genetic algorithm as a search strategy resulted to the selection of 44 features i.e. 82.81 % reduction. The classifiers examined in this paper were: decision trees, support vector machines, Neural Network, Logistic Regression, k-Nearest Neighbor. In order to estimate the accuracy of each classification model during the training phase, a stratified 10 fold cross validation was performed.

A. Decision Trees

The algorithm selected for decision tree building was C4.5, developed by R. Quinlan [17]. More specifically an open source implementation of the C4.5 algorithm in Weka known as J48 was utilized. The algorithm has a top down approach. It is a recursive divide and conquer algorithm. The training data are classified instances, while each one of these instances consists of features along with the class the specific instance belongs. One feature is selected as root node and the algorithm creates a branch for each possible feature value. That splits the instances into subsets, one for each branch that extends from the root node. The splitting criterion the algorithm uses is the normalized information gain. The feature with the highest normalized information gain is chosen to make the decision. Then the procedure is repeated recursively for each branch, selecting a feature at each node and only instances that reach that node are used to make the selection. This machine learning algorithm can be fine-tuned by setting up a lot of parameters. The parameters which were optimized in the experiment are shown on Table II.

TABLE II
PARAMETERS OF THE J48 LEARNING ALGORITHM

Parameter	Default Value	Chosen Value
Minimum number of instances per leaf	2	1
Use of unpruned trees	False	False
Confidence factor used for pruning	0.25	0.25
Consider subtree raising operation when pruning	True	True
Use of binary splits on nominal attributes	False	False

B. Support Vector Machines

A Support Vector Machine (SVM) is a machine learning method based on statistic learning theory. SVM try to find the maximum margin hyperplane that separates two classes. An adaptation of the LIBSVM [18] implementation was used in the following. Four types of kernel function linear, polynomial, radial basis function, and sigmoid are provided by LIBSVM. A Support Vector Classification (C-SVC) was used with Radial Basis Function (RBF) kernel. After various conducted experiments, it was found that the optimal value of gamma (G) parameter of the RBF kernel was 2.

C. Artificial Neural Network

A multilayer neural network using the backpropagation algorithm was implemented as a classifier in Weka. The neural network consisted of one hidden layer with 3 nodes. The number of inputs was the 44 selected features and the number of outputs the four possible categories namely jpeg, png, gif and pdf. The learning rate was set to 0.3 and in order to avoid local minimum and to accelerate the learning process, the momentum parameter was set to 0.2. The training time (epochs) after experimentation was set to 500.

D. Logistic Regression

The idea of Logistic Regression (LR) is to make linear regression produce probabilities. Instead of predicting classes, it predicts class probabilities. These class probabilities are estimated directly using the logit transform. In Weka the Logistic algorithm was utilized with the default parameter setup as shown on Table III.

TABLE III
THE DEFAULT PARAMETERS OF THE ALGORITHM

Parameters in Weka	
Maximum Iterations (MaxIts)	-1
Ridge Value (ridge)	1.0E-8

E. k-Nearest Neighbor

The k-Nearest Neighbor (k-NN) is a simple algorithm used for classification. The purpose of the k-NN algorithm is to use a training set - in which each one of instances is already classified- , in order to predict the classification of a new unknown instance in a test set. It is a lazy algorithm as it does not use the instances in training set to do any generalization. When a new instance is presented from a given test set, the algorithm searches the entire training set for the k most similar instances (the neighbors). To determine which of the k instances in the training set are most similar to a new input, a distance measure is used. The distance measure utilized in this implementation was the Euclidean distance. The output then can be calculated as the class with the highest frequency from the k-most similar instances. Each instance votes for their class and the class with the most votes is taken as the prediction. In order to find the optimum number of k, different implementations were done in Weka and it was found that the optimal value of k is 10.

ISSN 2395-8618

IV. Experimental results

Primarily, in order to evaluate the performance of every classifier used, 2208 unseen instances of unaltered files (equally distributed to the four categories as already shown on Table I) were presented to each classification model. The detailed accuracy by class, along with other performance metrics such as true positive rate (TP Rate), false positive rate (FP Rate), precision and recall for every one of the five classifiers are presented on tables IV-XII. Moreover, the resulted confusion matrix for each one of the learning algorithms examined, is presented.

TABLE IV
DETAILED ACCURACY BY CLASS USING DECISION TREE (J48)

Class	TP Rate	FP Rate	Precision	Recall
jpg	0.969	0.040	0.889	0.969
pdf	0.862	0.013	0.956	0.862
png	0.960	0.015	0.955	0.960
gif	0.991	0.004	0.989	0.991

TABLE V
CONFUSION MATRIX – DECISION TREE (J48)

Actual file type	Classified as			
	jpg	pdf	png	gif
jpg	535	9	4	4
pdf	57	476	17	2
png	10	12	530	0
gif	0	1	4	547

TABLE VI
DETAILED ACCURACY BY CLASS USING SVM

Class	TP Rate	FP Rate	Precision	Recall
jpg	1	0.014	0.960	1
pdf	0.953	0.001	0.996	0.953
png	0.986	0.009	0.973	0.986
gif	0.989	0	1	0.989

TABLE VII
CONFUSION MATRIX – SVM

Actual file type	Classified as			
	jpg	pdf	png	gif
jpg	552	0	0	0
pdf	17	526	9	0
png	6	2	544	0
gif	0	0	6	546

TABLE VIII
DETAILED ACCURACY BY CLASS USING NEURAL NETWORK

Class	TP Rate	FP Rate	Precision	Recall
jpg	1	0.002	0.995	1
pdf	0.987	0.002	0.993	0.987
png	0.993	0.005	0.984	0.993
gif	0.986	0.002	0.995	0.986

TABLE IX
CONFUSION MATRIX – NEURAL NETWORK

Actual file type	Classified as			
	jpg	pdf	png	gif
jpg	552	0	0	0
pdf	3	545	2	2
png	0	3	548	1
gif	0	1	7	544

TABLE X
DETAILED ACCURACY BY CLASS USING LR

Class	TP Rate	FP Rate	Precision	Recall
jpg	0.996	0.011	0.968	0.996
pdf	0.975	0.008	0.975	0.975
png	0.955	0.005	0.985	0.955
gif	0.987	0.005	0.986	0.987

TABLE XI
CONFUSION MATRIX – LOGISTIC REGRESSION

Actual file type	Classified as			
	jpg	pdf	png	gif
jpg	550	1	0	1
pdf	9	538	3	2
png	8	12	527	5
gif	1	1	5	545

TABLE XII
DETAILED ACCURACY BY CLASS USING k-NN

Class	TP Rate	FP Rate	Precision	Recall
jpg	0.993	0.018	0.950	0.993
pdf	0.942	0.007	0.979	0.942
png	0.975	0.006	0.982	0.975
gif	0.996	0.001	0.996	0.996

TABLE XIII
CONFUSION MATRIX – k-NN

Actual file type	Classified as			
	jpg	pdf	png	gif
jpg	548	4	0	0
pdf	23	520	8	1
png	6	7	538	1
gif	0	0	2	550

A. *The Digital Forensics Perspective*

In digital forensics it is very common someone to try to alter evidence, like by renaming image files to documents, in order to fool authorities. In order to examine if the proposed method identifies the correct file type when the file was altered, one third of the testing pdf files (168) was replaced with unseen image files whose extension and signature (magic bytes) was changed to pdf. The first test set contained 168 altered pdf files. These 168 altered files were actually jpeg images whose extension and signature was changed to pdf. Likewise, the 168 pdf files of the second dataset were actually png altered images and in the third data set the 168 pdf files

ISSN 2395-8618

were altered gif images. Therefore, three new test sets were created. Subsequently, unseen instances from all categories were presented to the models for evaluation. The resulted confusion matrix for every learning algorithm for each one of the three testing sets is shown on tables XIV-XVIII.

TABLE XIV
CONFUSION MATRIX – DECISION TREE (J48)

Forged file's Actual Type	Classified as			
	jpg	pdf	png	gif
168 jpg	167	1	0	0
168 png	8	3	157	0
168 gif	0	0	0	168

TABLE XV
CONFUSION MATRIX – SVM

Forged file's Actual Type	Classified as			
	jpg	pdf	png	gif
168 jpg	168	0	0	0
168 png	3	8	155	2
168 gif	0	1	0	167

TABLE XVI
CONFUSION MATRIX – NEURAL NETWORK (NN)

Forged file's Actual Type	Classified as			
	jpg	pdf	png	gif
168 jpg	168	0	0	0
168 png	0	2	166	0
168 gif	0	0	0	168

TABLE XVII
CONFUSION MATRIX – LOGISTIC REGRESSION (LR)

Forged file's Actual Type	Classified as			
	jpg	pdf	png	gif
168 jpg	168	0	0	0
168 png	7	6	150	5
168 gif	0	0	0	168

TABLE XVIII
CONFUSION MATRIX – K-NEAREST NEIGHBOR (KNN)

Forged file's Actual Type	Classified as			
	jpg	pdf	png	gif
168 jpg	167	1	0	0
168 png	8	3	157	0
168 gif	0	0	0	168

The combined confusion matrix for every classifier utilized in the experiments is shown on table XIX. The greyed color cells indicate the maximum accuracy achieved.

TABLE XIX
COMBINED CONFUSION MATRIX FOR THE FIVE CLASSIFIERS

Forged File types	**Prediction Accuracy (%)**				
	J48	**SVM**	**NN**	**LR**	**kNN**
jpg	99.40	▨ 100.00	▨ 100.00	▨ 100.00	99.40
png	93.45	92.26	▨ 98.81	89.28	93.45
gif	▨ 100.00	99.40	▨ 100.00	▨ 100.00	▨ 100.00

It is obvious from table XIX that even a very simple neural network achieved excellent results and identified extremely well almost all the forged files. The other classifiers achieved very high accuracy as well but we have to consider that in digital forensics the misclassification of even one file could be crucial in a court of law and could lead to the issue of an incorrect decision by the court members.

V. CONCLUSIONS

In this paper we examined a methodology for file type identification, which uses computational intelligence techniques for feature selection and classification. More specifically, this methodology was applied to the three most common image file types (jpg, png and gif) due to their significance to digital forensics. In order to ascertain the method's accuracy, different machine learning classifiers were utilized. A three stage process involving feature extraction (Byte Frequency Distribution), feature selection (genetic algorithm) and classification (decision tree, support vector machine, neural network, logistic regression and k-nearest neighbor) was examined. Experiments were conducted having files altered in a digital forensics perspective –by changing both their extension and signature- and the results were presented. The examined methodology showed -in most cases- very high and exceptional accuracy in file type identification, even if someone intentionally changes file's extension and signature. It was found that the classifier with the best results was the artificial neural network. In the future we plan to deploy the model, in fragments of files and examine its behavior. During our research we had strong evidence that the proposed method would work well too, although slight modifications and changes have to be made. Furthermore the correct identification of more file types should be another extension to our research and we plan to examine whether the proposed model depends on file compression.

REFERENCES

[1] K. Karampidis, G. Papadourakis, and I. Deligiannis, "File Type Identification -A Literature Review," in *9th International Conference on New Horizons in Industry Business and Education, NHIBE 2015*, 2015, p. 141.

[2] M. McDaniel, "Automatic File Type Detection Algorithm," James Madison University, 2001.

[3] M. McDaniel and M. H. Heydari, "Content based file type detection algorithms," *36th Annu. Hawaii Int. Conf. Syst. Sci. 2003. Proc.*, 2003.

[4] W. J. Li, K. Wang, S. J. Stolfo, and B. Herzog, "Fileprints: Identifying file types by n-gram analysis," *Proc. from 6th Annu. IEEE Syst. Man Cybern. Inf. Assur. Work. SMC 2005*, vol. 2005, no. June, pp. 64–71, 2005.

[5] J. Dunham, M. Sun, and J. Tseng, "Classifying file type of stream ciphers in depth using neural networks," in *The 3rd ACS/IEEE International Conference on Computer Systems and Applications*, 2005.

ISSN 2395-8618

[6] M. C. Amirani, M. Toorani, and a. a B. Shirazi, “A new approach to content-based file type detection,” in *IEEE Symposium on Computers and Communications*, 2008, no. July 2008, pp. 1103–1108.

[7] D. Cao, J. Luo, M. Yin, and H. Yang, “Feature selection based file type identification algorithm,” in *2010 IEEE International Conference on Intelligent Computing and Intelligent Systems*, 2010, vol. 3, pp. 58–62.

[8] I. Ahmed, K. Lhee, H. Shin, and M. Hong, “Content-based File-type Identification Using Cosine Similarity and a Divide-and-Conquer Approach,” *IETE Tech. Rev.*, vol. 27, no. 6, p. 465, Nov. 2010.

[9] I. Ahmed, K. Lhee, H. Shin, and M. Hong, “Fast content-based file-type identification,” in *7th Annual IFIP WG 11.9 International Conference on Digital Forensics*, 2011, pp. 65–75.

[10] M. C. Amirani, M. Toorani, and S. Mihandoost, “Feature-based Type Identification of File Fragments,” *Secur. Commun. Networks*, vol. 6, no. 1, pp. 115–128, Jan. 2013.

[11] J. D. Evensen, S. Lindahl, and M. Goodwin, “File-type Detection Using Naïve Bayes and n-gram Analysis,” *Norwegian Information Security Conference, NISK*, vol. 7, no. 1. Fredrikstad, 2014.

[12] K. Karampidis and G. Papadourakis, “File Type Identification for Digital Forensics,” Springer International Publishing, 2016, pp. 266–274.

[13] N. S. Alamri and W. H. Allen, “A comparative study of file-type identification techniques,” in *SoutheastCon 2015*, 2015, pp. 1–5.

[14] M. Hall, “Correlation-based feature selection for machine learning,” The University of Waicato, 1999.

[15] L. Fei-Fei, R. Fergus, and P. Perona, “Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories,” p. 178.

[16] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, “The WEKA data mining software,” *ACM SIGKDD Explor. Newsl.*, vol. 11, no. 1, p. 10, Nov. 2009.

[17] S. L. Salzberg, “C4.5: Programs for Machine Learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993,” *Mach. Learn.*, vol. 16, no. 3, pp. 235–240, Sep. 1994.

[18] C.-C. Chang and C.-J. Lin, “LIBSVM: A Library for Support Vector Machines,” *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 27:1–27:27, 2011.