An Enhanced Multiclass Support Vector Machine Model and its Application to Classifying File Systems Affected by a Digital Crime

Article *in* Journal of King Saud University · October 2019

DOI: 10.1016/j.jksuci.2019.10.010

1 author:

Rami Mustafa A Mohammad
Imam Abdul Rahman bin Faisal University
14 PUBLICATIONS **357** CITATIONS

Journal of King Saud University – Computer and Information Sciences xxx (xxxx) xxx

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

An Enhanced Multiclass Support Vector Machine Model and its Application to Classifying File Systems Affected by a Digital Crime

Rami Mustafa A. Mohammad

Department of Computer Information Systems, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam, Saudi Arabia

ARTICLE INFO

Article history:
Received 21 July 2019
Revised 5 October 2019
Accepted 26 October 2019
Available online xxxx

Keywords:
Digital-forensics
File-systems
SVM
Log-Files
Digital-evidence

ABSTRACT

The digital revolution we are witnessing nowadays goes hand in hand with a revolution in cybercrime. This irrefutable fact has been a major reason for making digital forensic (DF) a pressing and timely topic to investigate. Thanks to the file system which is a rich source of digital evidence that may prove or deny a digital crime. Yet, although there are many tools that can be used to extract potentially conclusive evidence from the file system, there is still a need to develop effective techniques for evaluating the extracted evidence and link it directly to a digital crime. Machine learning can be posed as a possible solution looming in the horizon. This article proposes an Enhanced Multiclass Support Vector Machine *(EMSVM)* model that aims to improve the classification performance. The EMSVM suggests a new technique in selecting the most effective set of parameters when building a SVM model. In addition, since the DF is considered a multiclass classification problem duo to the fact that a file system might be accecced by more than one application, the EMSVM enhances the class assignment mechanism by supporting multi-class classification. The article then investigates the applicability of the proposed model in analysing incriminating digital evidence by inspecting the historical activities of file systems to realize if a malicious program manipulated them. The results obtained from the proposed model were promising when compared to several machine-learning algorithms.

1. Introduction and motivation

Computers, networks, and the Internet have turned into fundamental parts of our everyday financial and social activities. Individual users are not the only beneficiaries of these technologies, but organizations have also benefited from it for achieving their ultimate goals and increase their values. For instance, organizations that provide online and cloud-based services have a greater opportunity to secure a competitive advantage over their opponents by serving clients all over the world (Mohammad and AbuMansour, 2017). Undeniably, the Internet facilitates reaching and serving customers around the world without any marketplace limitations and with feasible usage of ecommerce. Nowadays, the number of consumers who make online trading has increased significantly, and the number is still increasing as an inevitable result of the technological improvements. Billions of dollars are constantly exchanged electronically. This huge amount of cash allures fraudsters to launch their dishonest campaigns. Consequently, computer and Internet clients might be prone to several categories of risks that could result in brand reputation damage, loss of private information, identity theft, financial losses, and most importantly loss of clients' belief in ecommerce and web-based services or in the worst-cases loss of confidence in all new technologies which in the first place were invented to facilitate our daily life. Thus, the validity of the computers and the Internet for business and personal processes becomes questionable (Mohammad and AbuMansour, 2017). In general, despite the fact that the new technologies may have amazing potential for personal or organizational gain, it may also be used as an effective way to conduct several cybercrime attacks.

In 2018, approximately 700 million persons were victims of at least one type of cybercrimes attack (Bera, 2019). Nevertheless, only 10% of incidents are actually reported to the competent authorities (Powell, 2019). the because subjects may feel shy, ashamed, or simply because they think that there is no deterrent action that can be taken against these criminals. Additionally, the

Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

E-mail address: rmmohammad@iau.edu.sa

https://doi.org/10.1016/j.jksuci.2019.10.010
1319-1578/

availability of qualified and experienced personnel required by the government bodies to mitigate computer misuse offences remains debatable in spite of the determined initiatives of several governments. As an example, in 2006 the former U.S. president "George W. Bush" has initiated the "Identity Theft Task Force" (Executive Order 13402, 2012) which is intended to make sure that efforts from the federal authorities will be more beneficial, effective, and applicable in protecting against and discovering cybercrime incidents. A decade ago, a collaboration agreement was signed between the Australian authorities and Microsoft where the later one will train the law enforcement officials about the most effective ways to deal with different kind of cybercrimes (Mohammad et al., 2015). The United Kingdom have in turn strengthened its legal arsenals to protect against fraud and cybercrime incidents by enacting penalties of 10 years in prison.

Various areas of study have emerged in the last decade in response to the high rate of cybercrime. Digital forensics (DF) is one of these areas. Typically, DF investigations defined as a precise procedures that utilize scientific standards and reliable tools to identify, gather, maintain, document, validate, and analyze information stored on electronic devices' main memory and auxiliary storage media that are directly related to a suspicious event with the aim of realizing the damage caused by this event, the sequence of the event, the action that triggered the event, and who performed the event. These suspicious events have dramatic consequences on both individuals, and financial institutes. At the individual level, the result of these suspicious events may be the theft of private information and financial IDs. At the institutions level, the financial position of the organization may deteriorate as a result of leakage of business secrets and this will certainly harm the image and reputation of the organization.

DF procedures are usually initiated in response to several types of electronic crimes, among others we name e-terrorism, intellectual property theft, money laundering, data abuse, unauthorized access to private information, illegal access to computer systems, etc (Nelson et al., 2016). Generally, DF investigations aim to identify the committers of a digital crime by recognizing, collecting, analyzing, and reconstructing the crime related events to confirm or deny the committers relation to the crime using reliable and reproducible evidence and then presenting them in an acceptable form to the court.

All kind of digital storage media, i.e. USB memory sticks, hard drives, magnetics strip, ZIP diskette are subject to inspection during the forensics analysis no matter whether they exist in mobile devices, digital cameras, network equipment, or personal computers. As an inevitable consequence, the amount of data that can be collected when investigating electronic crime scenes will increase dramatically. However, the real value of the collected data can be validated by its capability to produce accurate information that facilitate decision making. Typically, DF investigations involve conventional manual activities where skilled investigators use their experience to produce reports describing the data insight using some tools and submit it to the court. Yet, understanding the data in its originally collected form could be a painstaking task for the DF investigators. Consequently, some useful information might be lost. Therefore, some tools such as LastActivityView (NirSoft), Sleuth Kit (Sleuth Kit), and SANS SFIT (SANS DFIR) are normally used to transform the collected data into a more understandable form. As soon as the data transformation is accomplished, some statistical tools can be used to extract and derive data of interest. Nevertheless, the manual analysis process may deteriorate and may produce inaccurate and sometimes unrealistic results if the size and dimensionality of the collected data intensely expandeds. Investigation costs is another reason why manual analysis might not be the right analysis option. It is important here to draw attention that investigation costs include both financial costs and time costs. More importantly, these days investigators may face very complex incidents that require searching for interlocking threads within the collected data to obtain some comprehensive evidence, and that is another reason for not relying on manual analysis. One of the main issues that a digital forensic investigator might face is whats called Scope Creep (Nelson et al., 2016) which is direct result of the huge amount of data that an investigator needs to study. Therefore, investigators may resort to more reliable and efficient methods for digging into the collected data in order to obtain conclusive and comprehensive evidence when data analysis goes beyond the capabilities of conventional manual analysis. Solutions hovering around Data Mining (DM) and Machine Learning (ML) loom as promising solutions. DM and ML have proved their capabilities in many domains as effective methods in providing accurate information for decision makers (Mohammad et al., 2013b; Khemphila and Boonjing, 2011; Lee and Xiang, 2001). DM and ML can be defined as *("the process of exploring data from several perspectives and converting them into meaningful forms")* (Kaufmann et al., 2011).

There are several sources where discriminative evidence can be gathered, for instance temporary files, cookies, web browser history-files, and log files. However, system files remain the most valuable source of critical digital evidence. Yet, the file systems can be easily modified whether initially for the purpose of hiding some evidence or unintentionally by the routine computer systems operations.

Recognizing the system files influenced by a suspicious incident is an essential step to confirm or deny the accused's relationship to the incident. This will in deed help in identifying the set of system files that should be used in event reconstruction. Even reconstruction is very helpful in realizing how the incident is actually conducted and hence comprehend any weaknesses in the system and then respond by resolving such weaknesses in order to avoid future similar attackes. Metadata accompanying system files is also a useful source of information for digital crime investigators. The problem lays in that several applications may manipulate the same system files at different timespan. Hence, it would be very crucial to know the precise sequence of events that affected the system files so that the investigators can decide whether a particular activity corresponds to a reliable or to a malicious application program.

The current research utilizes the concepts of Support Vector Machine (SVM) for creating a new model that can be used for analyzing the system file activities to realize if they were accessed by a particular software application. The proposed model pays attention to adequately searchs for the best parameters in order to achieve the best classification results. In addition, the model aims to identify the system files that will indeed contribute in rebuilding a digital crime events. The performance of the recommended model is going to be contrasted to several other DM and ML techniques.

SVM has been and continues to be the focus of many researchers because of its ability to generalize even with limited training dataset or with huge training dataset. In other words, normally SVM models do not suffer from the dimensionality dilemma which considered a common problem encountered by most DM and ML approaches. SVM has fundamental theoretical reference to statistical learning theories in the sense that it applies its robust phenomenon known as *("kernel trick")* which is capable to transform non-separable dataset into separable one that has clear dividing margin between different class values. There are several other reasons that prompted us to employ the SVM, the most important of which is that this algorithm proved to be efficient in many fields

Please cite this article as: R. M. A. Mohammad, An Enhanced Multiclass Support Vector Machine Model and its Application to Classifying File Systems Affected by a Digital Crime, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.10.010

and outperformed many other DM and ML algorithms in terms of different comparative criteria (Qiu et al., 2020; Wu et al., 2020; Dongil et al., 2020).

In this study, an enhanced SVM *(EMSVM)* model is created and tested using MATLAB environment (MathWorks, 2019). Such an algorithm followed a systematic method in searching for the parameter values that would produce better classification accuracy. The experimental results showed that the suggested algorithm surpasses all DM and ML algorithms that are considered in this study. These results were not achieved coincidentally but because of the efficient way in which the proposed algorithm is built. Overall, the proposed algorithm gave the opportunity to SVM algorithm to add the file system forensics to its long list domains where it can successfully achieve superb classification results.

The remaining parts of the study is ordered as below:

Section 2 offers a background discussion that links earlier works to the work presented in this article. The proposed model is illustrated in Section 3. The same section elaborates the procedure employed for selecting the parameters of the suggested model. Moreover, this section shows how the recommended model deals with multi-class classification domains in the sense that the domain that is counted in this research is a good example of multi-class classification. Of course, a number of experiments were carried out to assess the applicability of the *EMSVM*. Such experiments were depicted in Section 4. The obtained results were detailed in Section 5. Lastly, the conclusion were presented in Section 6.

2. Backgrounds and literature review

Generally, there are many signs that usually indicate whether a pc is a victim of cybercrimes. Such signs if occurred encourage the starting of DF investigations. Indications associated with cybercrimes incidents might include:

- Pc boot-up extremely slow
- Pc requires reasonable length of time to shutdown
- Slow response
- Lost or perhaps ruined data
- Strange processe(s) running in the task-manager
- Sudden low storage space alerts
- Undesirable pop-up warnings which might propagate spiteful web links
- Violence alerts
- Laptop battery emptying quickly
- Unstable wireless connection
- Showing the "Blue Screen of Death (BSoD)" several times in short timeframe.

Typically, DF investigations aim to answer the five major questions those are:

1- Is there any suspicious activity that requires starting the - investigation?
2- What was the motives behind the incident?
3- Which files and folders were affected by the attack?
4- When the attack actually began?
5- How the crime was committed (crime weapon)?

Existing researches in the field focus on gathering up and inspecting evidence obtained from assorted sources that might be able to come up with clear-cut answers to these questions. Though, system files received remarkable attention as a result of the dependable details they may offer in comparison to other possible sources of evidence (Cho and Rogers, 2011).

In 1995, Pollitt founded the basic principles for examining different possible computer evidence (Pollitt, 1995). Pollitt recommended a 4 stages procedure for pinpointing digital evidence. These stages were driven from the procedure of submitting documented evidence in courts and they involve "Acquisition, Identification, Evaluation, and Admission". A roadmap for digital forensics analysis is recommended in 2001 (Palmer, 2001). The roadmap is deemed as the basis for most advised models which implemented till now and it includes "Identification, Preservation, Collection, Examination, Analysis, Presentation, and Decision".

A model which is mainly influenced by (Palmer, 2001) is proposed in (ClintCarr and Gunsch, 2002) and is referred to as "Abstract Digital Forensics Model". The model has added 2 more stages to the stages recommended in (Palmer, 2001) these are "Preparation and Approach strategy". These stages are going to be fulfilled somewhere between the first two stages. The first added stage is aimed at setting up equipment and tools that usually considered useful in addressing and analyzing cybercrimes. on the other hand, the second added stage is aimed at formulating the analysis methodology depending on the recognized effect on spectators and the engaged technologies. However, the key weaknesses of the model as claimed by the model creators is that it is very general and above all there is no precise strategy for evaluating it.

Looking at the large picture of the stages recommended in (Palmer, 2001) and (Pollitt, 1995) that are regarded as the backbone of almost all succeeding studies, we can clearly realize that the stages of doing a DF analysis adhere to stages of creating/developing a DM and ML models. Nevertheless, the majority of such studies agree with the fact that the event rebuilding procedure is definitely an important task for every successful DF investigation. That in turn demand deciding on the most decisive group of evidence.

The researchers in (Henry et al., 2007) claimed that event rebuilding procedure begins by collating decisive evidence that are normally utilized for forming an initial assumption about the incident. The actual DF analysis is then begins by creating hypotheses about the incident. These hypotheses will after that be assessed and analyzed. Finally, conclusions will be deduced.

The researchers in (Chabot et al., 2014), recommended demonstrating the system behaviors as "a Finite State Machine". Consequently, action rebuilding normally regarded as a procedure of looking for sequence of changes that match the constraints enforced by the evidences. An innovative method for actions rebuilding which is principally depending on identifying convicting data objects and the associations amongst such was suggested in (Rynearson, 2002). Data objects comprise relational, functional, and temporal association with several events.

Ontology based technique has efficiently applied for actions rebuilding. As an example, the researchers in (Schatz et al., 2004) recommended gathering up and retaining the events and then by applying ontology technique they will be capable to rebuild the previous events. In this approach, any modification of objects with time is depicted as entities. Yet, a primary challenge in such technique is how to represent the time model since the researchers utilized instants instead of intervals. The research carried out in (Zhu et al., 2009), suggested utilizing the "Shellbag info" residing in the Windows registries. Shellbag holds user's personal preferences including window location, windows specifications, folders location, etc. Moreover, they offer details about folders and files even after they happen to be updated, deleted, or moved. One more approach which makes use of computer registries is offered in (Carvey, 2005). The researchers recommended making use of the info kept in "NTUSER.DAT" due to the fact that such file includes

Please cite this article as: R. M. A. Mohammad, An Enhanced Multiclass Support Vector Machine Model and its Application to Classifying File Systems Affected by a Digital Crime, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.10.010

all users' configurations kept in computer registries. Currently, many free and commercial utilities have been developed to help collating actions and keep them in a repository since it could be hard to assess the gathered evidence if they were offered in their original format.

Typically, the attacker might make an effort to protect his footprints through camouflaging several vital files through altering their types. For instance, swapping a file type from text to image. In 2003, an innovative technique was constructed using SVM for identifying file types through analyzing the structure inside their content (DeVel, 2003). SVM has also correctly employed for distinguishing anomalous actions in Windows registries (Carrier and Spafford, 2005). Nevertheless, the research showed that results attained from "Probabilistic Anomaly Detection Algorithm" beats the result obtained by SVM.

Obviously, textual docs as well as emails regarded as a valuable source of evidence when doing a forensics investigation. There are almost 3 billion email account can be found world-wide. Yet, corporate-based email accounts constitute more than 25%. For every corporate email more than 100 email messages are usually dispatched and delivered on a daily basis. Such figures certainly indicate that email is actually an essential way to communicate, and therefore looked at as a possible source of evidence that usually cannot be omitted. When working with email messages, an essential activity may be the authorship verification and attribution. Many researches have tackled this issue through investigating both the email document structure *(*i.e. *"email header", "number of paragraphs" and "number of lines", etc.)* as well as linguistic patterns *(*i.e.*"vocabulary richness", "occurrences of punctuations", "character count", etc.).* In (Vel et al., 2001) SVM has showed its abilities once again in the sense that it obtained accuracies rage from 84 to 100% in a data set contains 156 emails.

In general, the roadmap of the researches that lead to start making use of DM and stimulated us for conding this reasacrh can be suberized in Table 1.

3. Enhanced multiclass SVM (EMSVM) model

This section the EMSVM is described comprehensively. Fig. 1 shows the key phases of building the proposed EMSVM and in the next subsections these steps are explained in detail.

3.1. Modeling EMSVM classifiers

SVM is deemed a statistical-based ML method with distinctive capacity to model complicated relationships between variables. SVM can superbly combines the generalization ability with the capability to handle the curse of dimensionality. Normally, curse of dimensionality negatively affects the performance of DM and ML algorithms (Mohammad and Alqahtani, 2019). Yet, SVM has distinguished itself as a talented technique that is able to perform outstandingly even if few examples were used for training the algorithm. SVMs are exceptionally enabled by kernel functions to efficiently map non-linearly separable problems into higher dimensions so that they can be easily separated. Kernel mapping offers a unified framework for the majority of commonly utilized models. Mapping non-linear separable examples *("input space")* into separable ones *("feature space")* that can be easily separated is done by transforming the initial dimension-space of the training dataset to higher dimension-space. In fact, SVM was developed primarily for classification domains but it has also proved its ability and efficiency in regression domains. The case investigated in this research is deemed a classification problem. Though, it is well known that the more the classification model is capable to maximize the margins that separate the classes the more generalized the model will be (Witten et al., 2011). In SVM, generalization is usually attained by creating a set of vectors that could be sparse but at the same time decisive in separating one class from another (Gonsalves et al., 2019). The examples that lie on the vectors' boundary encapsulate the information needed to separate the classes and hence can be used later for classifying unseen examples. Fig. 2 illustrates how a margin is capable to separate the class value between 2 groups of data using SVM.

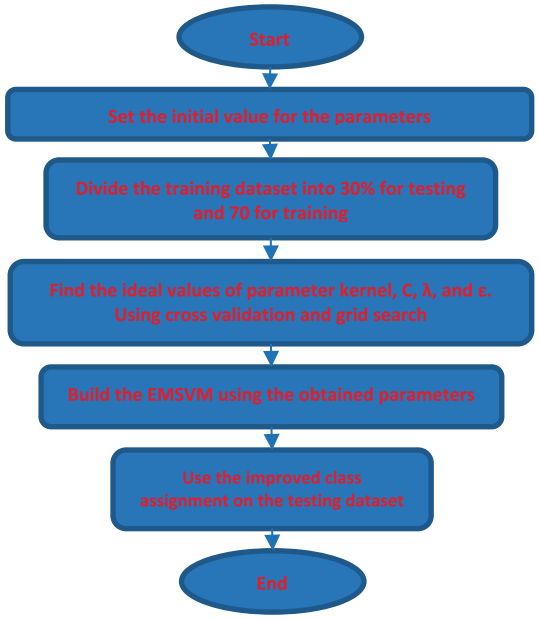

Fig. 1. The main phases of building the EMSVM.

Table 1
Roadmap of researches lead to using DM techniques for DF.

Refs.	Contribution
(Pollitt, 1995)	The basic principles for examining different possible computer evidence
(Palmer, 2001)	A roadmap for digital forensics analysis
(ClintCarr and Gunsch, 2002)	Adding 2 more stages to the stages recommended in (Palmer, 2001)
(Henry et al., 2007)	Event rebuilding procedure begins by collating decisive evidence
(Chabot et al., 2014)	Demonstrating the system behaviors as "a Finite State Machine"
(Schatz et al., 2004)	Using ontology based technique
(Zhu et al., 2009)	Using the "Shellbag info" residing in the Windows registries
(Carvey, 2005)	Making use of the info kept in "NTUSER.DAT"
(Carrier and Spafford, 2005)	Using SVM for identifying file types

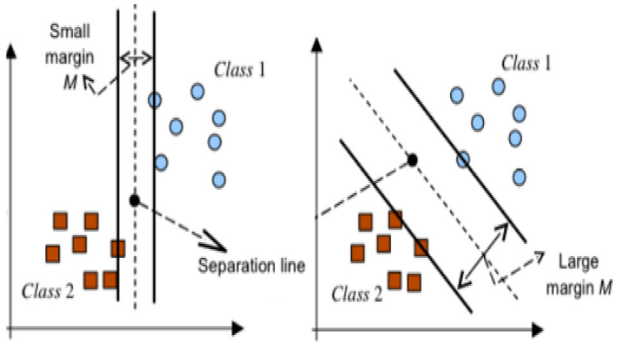

Fig. 2. Margin Creation in SVM.

Please cite this article as: R. M. A. Mohammad, An Enhanced Multiclass Support Vector Machine Model and its Application to Classifying File Systems Affected by a Digital Crime, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.10.010

In general, in any classification problem the most important goal is to identify the relationship amongst the set of input variables and class variable(s) of a specific training dataset $T=\{X,Y\}$ where $X \epsilon R^s$ denotes the *n-by-m* matrix of *n* input features also known as independent variables or predictors and *m* examples also known as training instances. It is actually worthy to note that $Y \epsilon R$ in classification problems, yet in regression problems $X \subseteq R$. To illustrate further, let assume that there is a classification problem that has a training dataset $T=\{x_{ij}, x_{i+1j+1},.., x_{nm}, c_1, c_2, \ldots c_t\}$ where $i = 1,2,\ldots,n$ represent the input features, $j = 1,2,\ldots,m$ represent the examples, and *c* represents the class value(s) in a training dataset that have *n* input features, *m* examples and *t* class values where *t greater than 1*. The training dataset is used for creating classification models *("in our research using an enhanced SVM model")* that map the input variables $X \epsilon R^s$ into high dimensional feature spaces $I\!J \epsilon H$, and creates an *"Optimal Separating Hyperplane (OSH)"* that minimizes the margin *("the distance between the hyperplane and the closest data point of every class H")* using equation (1).

$$sgn\left(\sum_{i=1}^{n} y_i \propto_i \cdot K(x_i.x_j) + b\right) \tag{1}$$

where $x_j = 1,2,\ldots,Z$ are the so-called *"Support Vectors (SV)"*. The coefficient $\propto_i$ as well as the bias *b* are attained using *"lagrange dual equation"* which is shown in Eq. (2) and also known as *"convex quadratic programming (QP)"*.

$$MAX\left(\sum_{i=1}^{n} \propto_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \propto_i \propto_j \cdot y_i y_j \cdot K(x_i.x_j)\right) \tag{2}$$

where:

$$\sum_{i=1}^{n} \propto_i y_i = 0 \tag{3}$$

Here, x_j is called support vector only if $0 \leqslant \propto_i \leqslant C$. Where *C* is the regularization factor that controls the tradeoff between misclassification error and the margin. In other words, *C* regulates the tradeoff cost between minimizing the model's complexity and minimizing the error rate of the training process. It should be mentioned here that if *C* is too large then the SVM might produce an over-fitted model in the sense that the SVM have a high penalty for the non-separable points. Yet, a small *C* value might produce an under-fitted model. However, *K* is the kernel function that is responsible for converting the dataset into hyperplanes.

There are several kinds of kernel function that can be used in SVM. The most commonly used ones are "Polynomial" and "Radial Basis Function (RBF)". The polynomial function of degree *d* is calculated per Eq. (4):

$$K(x_i.x_j) = \left(x_i^T \cdot x_j + 1\right)^d \tag{4}$$

Its worth mentioning that the polynomial function reverts to linear function if $d = 1$ and in this case the additive constant in the equation is ignored. Hence, the linear kernel function can be calculated per Eq. (5):

$$K(x_i.x_j) = x_i^T x_j \tag{5}$$

On the other hand, RBF (*also known as Gaussian kernel)* is calculated as shown in Eq. (6):

$$K(x_i.x_j) = \exp\left(-\gamma x_i - x_j^d\right) \tag{6}$$

Where γ controls the Gaussian width. In other words, it plays the same role as *d* in Polynomial kernel in the sense that it controls the flexibility of the produced classifier. Here, $\gamma = \frac{1}{2\sigma^2}$ and σ is a free parameter.

One more kernel function that is related to Neural Networks is called sigmoid kernel. This kernel function was firstly used in 1995 (Cortes and Vapnik, 1995) and is calculated using equation (7) below:

$$K(x_i.x_j) = tan\ h\left(\gamma x_i^T x_j + r\right) \tag{7}$$

where γ and *r* are kernel factors.

3.2. Parameter settings in EMSVM

When building any classification model and in order to achieve a valid training and after that rational testing of the produced classification model(s), there is a need to research and find the ideal values of the most vital parameters. Given that a SVM model possess vitally important parameters that have irrefutable impact on the model's overall performance as well as its classification capability with respect to the classification problems at hand, a scientific parameter exploration strategy is employed in the present study to make sure that the ideal parameter values are actually picked. In general, when constructing any SVM model the most important parameters that should be carefully selected are *C, "bound on the Lagrangian multipliers"*, λ, *"conditioning parameter in QP", K, "The kernel"*. Another parameter that should be taken into account that is ε *("Epsilon")* which defines the margin of tolerance where no penalty is given to errors. Normally, these parameter are selected using trial and error method which is a painstaking and overwhelming process (Tzu-Liang et al., 2015; Jin et al., 2014; Lai et al., 2008). In this study, all the above mentioned parameters needed for constructing a robust SVM model are adjusted by using a *"Cross-Validation"* over the offered dataset.

The method works as follows:

for every run of produced training and testing sets, the obtained accuracy values *("percentage of accurately labeled examples")* are saved for parameters *C*, λ, *and* ε. Digging within all available values of the parameters within a given range can determine the ideal performance measures as well as the related values of the parameters for the fixed set of features. This procedure is redone for each SVM kernel on hand, every time by using a gradual step of parameters. Hence, the ideal values of the parameters and the kernel alternative associated with the best overall performance measure are determined. Overall, the steps of the proposed parameter searching method are explained hereafter:

Step 1: Select the initial kernel from the list of possible kernels, i.e. Polynomial, RBF, Linear, Gaussian, and Sigmoid.
Step 2: Using "Cross-Validation-Technique" find the ideal values of parameter C, λ, and ε, then save the related measure i.e. accuracy. Here, grid search algorithm (Bergstra and Bengio, 2012). is used for identifying the ideal kernel's parameters using the overall accuracy in $\log_2$ space. Firstly, a 2-fold cross validation is utilized for parameters selection. Then, the ideal point in the space is considered as a center point and 10-fold cross validation is applied with the adjunct parameters. If better parameter values are obtained, they will act as new center and 10-fold cross validation is applied again. This step should be repeated over and over again until no better parameters are obtained, or until the parameters are at the border of the grid.
Step 3: If there is any alternative kernel still available then add it and go to step 2. Otherwise, move to step 4.
Step 4: Find the best obtained performance measure and its corresponding kernel and parameter settings.
Step 5: Train the final model by emplying the kernel and the parameter settings found in Step4.
Step 6: Use the SVM created in Step 5 for classifying unseen dataset examples.

Please cite this article as: R. M. A. Mohammad, An Enhanced Multiclass Support Vector Machine Model and its Application to Classifying File Systems Affected by a Digital Crime, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.10.010

3.3. Class assignment in EMSVM

In DM and ML, multinomial or multiclass classification is the situation of categorizing a dataset example as one of three or more possible classes. However, classifying examples as one of two classes is known as binary classification. Multiclass classification must not be confused with multi-labels classification, in which multi-labels can be forecasted for every example. In the DF classification for example there could be overlaps amongst the file-system activates. For example, a lot of the file-system happen to be shared amongst multiple applications. Therefore, some system files might be associated with several applications. Although some classification methods typically enable the utilization of even more than two classes, others happen to be naturally binary methods. SVM is regarded as one of such methods (Nazari and Kang, 2015; Ma and Chen, 2015). Yet, SVM can be transformed into multiclass classifiers by using a number of techniques. Probably the most widely used techniques is the "One-vs-Rest" approach (Rocha and Goldenstein, 2014). This method includes training a single classifier for every class, considering the examples of this class being positive examples and all other kinds of examples as negative ones. This tactic necessitates the base classifiers to generate a real-valued confidence rate for its decision, instead of simply producing a single class label. Nevertheless, the standard one-vs-rest multiclass classification SVM model might result in the so-called the rejection area dilemma leading to overall performance deterioration of the accuracy. To resolve such an issue, a kernel metric-based technique is recommended throughout this research.

Considering the case shown in Fig. 3. In the event of merely a single decision function (*DS*) is certainly acceptable, the examples might fall within area A, consequently regular classification shall be performed. In cases where none or multiple of the *DS* are acceptable, the example are going to fall within the rejection areas B or C for at least one of the *DS*. Further, in the event that multiple *DS* are acceptable, the examples are going to fall within rejection area B for at least one of the *DS*. In the case that all *DS* are not acceptable, the examples are going to fall within the rejection area C. Whenever the examples fall under any rejection area, the standard one-against-rest technique might not work. In order to handle this condition, the computed space distance between examples and related *DS* are employed for classifying such examples.

For instance, if example q^* falls under rejection area B, the distance $s(q^*)$ between q^* *and the t-th* best acceptable hyperplane must be determined. Such distance is calculated as per Eq. (8).

$$s_t(q^*) = \frac{|D_t(q^*)|}{\Omega_t} \tag{8}$$

where Ω_t signifies the module of the normal vector of *t-th* best hyperplane. Finally, the *DS* of the *t-th* best hyperplane is calculated as shown in Eq. (9).

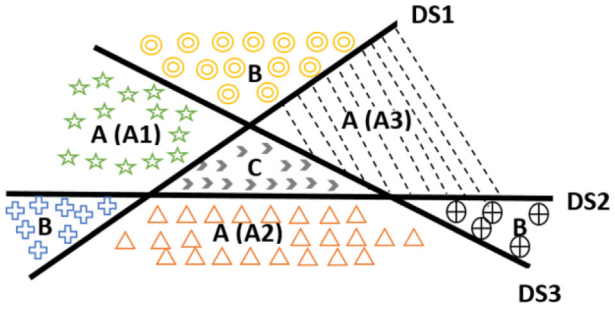

Fig. 3. Example of multi-class classification.

$$DS_t(q^*) = \sum_{z=1}^{n^t_{sv}} \lambda_{ts} y_{ts} K(q_{ts}.q^*) + b_t \tag{9}$$

where n^t_{sv} signifies the amount of SVs in the *t-th* best hyperplane, λ_{ts} denotes the "lagrange multiplier" of the *s-th* SV in the *t-th* best hyperplane, and $1 \leqslant s \leqslant n^t_{sv}$ Also, q_{ts} is the eigenvector and y_{ts} is the class value of the *s-th* SV in the *t-th* best hyperplane. However, $K(q_{ts}.q^*)$ denotes the value of the kernel function between q^* and q_{ts}. In addition, b_t is the deviation of the *t-th* hyperplane. Overall, the nearer the example is to the decision surface the higher the wrong classification will be produced. Hence, an example should be categorized into class with the higher distance using Eq. (10).

$$q^* \epsilon \arg(max_t(s_t)) \tag{10}$$

On the other hand, if the example falls under rejection area C, the distance between the example and all hyperplane(s) should be computed. The nearer the example is to the decision surface the higher possibility the example fitting to the related class of the other side of the surface. Consequently, the example must be categorized into the class with minimum distance of the rejection area using Eq. (11).

$$q^* \epsilon \arg(min_t(s_t)) \tag{11}$$

4. Experiments evaluation methodology

The following set of tests is going to evaluate the capacity for the recommended EMSVM model for classifying which files happen to be altered by a particular software application by examining the characteristics (fingerprints) any application creates once opening, changing, altering, or removing a particular file-system. Identifying the group of files impacted by an event might help rebuilding earlier incidents. However, event rebuilding procedure is deemed as one potential research area. It should be mentioned here that the model was implemented using written program in MATLAB 2019a (MathWorks, 2019) environment. The experiments have been completely carriedut in a computer that have Pentium Intel® CoreTM i5, 2.40 GHz processor, and a 4 gigabyte memory. The installed OS was 64-bit Windows 7. The next su-sections (i.e. sub-section 4.1 until 4.8) describe the methodology followed in evaluating the EMSVM.

4.1. Identifying the input features

This phase offers benefits of decreasing the time needed for developing DM and ML models as well as reducing the storage requirement. Additionally, this technique assures the compactness and the simplicity of the developed model. Moreover, it aimed at picking a group of input attributes could be the most important attributes in forecasting the value of the class variable(s) also known as *"output feature(s)"*. When the dataset dimensionality is decreased that would indeed results in saving the memory and the time needed for training the DM and ML model. Several attributes is often obtained from *"file system"*, *"audit logs entries"*, as well as *"registry information"*. The full set of viable input attributes which may be utilized in file system based DF investigation are reviewed in depth in a published book titled *"File System Forensic Analysis"* which is released in 2005 (Carrier, 2005). Applying all these attributes can result in a high-dimension dilemma also known as *"the curse of dimensionality"*. For that reason, the group of attributes which is proved from the past researches (Carvey, 2005) as vital ones are utilized in our study. Additionally, the *"Information Gain (IG)"* technique is utilized for selecting the most important group of attributes from available attribute set. IG is regarded as a popular method to assess how good the features

Please cite this article as: R. M. A. Mohammad, An Enhanced Multiclass Support Vector Machine Model and its Application to Classifying File Systems Affected by a Digital Crime, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.10.010

are in predicting the class value(s) so it is applied to many classification fields (Khemphila and Boonjing, 2011; Mohammad, 2016; Basnet et al., 2012; Lee and Xiang, 2001). The set of input features considered in this study are shown in Table 2. In the next section, several scenarios will be used for forming the training dataset.

4.2. Collecting dataset examples

A training dataset comprises a number of fingerprints *("features or attributes")* associated with file-system events *("file system metadata")*, PC registries, as well as system event logs has been gathered. These 3 sources are regarded as the main sources for identifying cybercrime related evidence (Atif and Ruighaver, 2002). A potential benefit behind incorporating the attributes coming from such 3 sources is the fact that in case one attribute is ruined or missing in any source it might however be existing in another source. The case study reviewed through this research is regarded as a supervised classification issue that demand the existence of class value(s) which is the "name of software application" in our case-study. Different software applications have been started in different time-periods. Then the set of attributes related the file-systems that have been altered by such programs are gathered. These attributes will be represented as an array. A tool was developed by making use of *("FileSystemWatcher")* library in *(" dot net framework 4.6.2″)* in order to collect the attributes related to the file-system that has been modified by several software applications. Such a tool monitors and tracks file-system modification actions including the time the file has been created, changed, or accessed. A structured repository is then used for storing the collected attributes. Several scenarios were prepared in order to collect the training dataset examples. Every scenario tries to access file-system in various methods. In the current research study, virtual machine "VMware" (Greene et al., 1998) is employed for gathering the training dataset examples. One primary aim from employing "VMware" is to provide fresh-state operating systems (OS), hence, reducing the effects of unrequired applications. The OS employed for gathering the dataset is "MS Windows 7″. Selecting this OS is because of its wide-spread utilization. In addition, it is the most frequently targeted OS (Zhu et al., 2009).

For the purpose of this research and in order to collect the training dataset examples, twelve different software applications were considered. These software applications were selected in the first place to ensure the functional variety. The software applications nominated and used in this research are: *"MS-Power Point, MS-Excel, MS-Access, MS-Word, Photoshop, Adobe Acrobat Reader, Chrome, Eclipse, Windows Movie Maker, Weka, OneDrive, Dropbox"*. Such software applications were ran in four different scenarios.

Scenario-I: This scenario is the simplest scenario amongst others. The considered software applications were ran separately one after another. As soon as each particular software application is completely executed, it will be closed without doing any extra operations, i.e. creating new file, opening a saved file, starting a video, browsing, etc.

Scenario-II: Again, the software applications were ran one after another. Yet, in this scenario, as soon as the software application is started, only one task will be executed. For instance, opening a saved file, browsing a single webpage, watching a saved video, etc. However, as in scenario-I, the software applications will be executed separately, i.e. one after another and no two software applications can be executed simultaneously. The moment that a particular application is loaded and the intended task is executed it will be closed and the next one is started.

Scenario-III: The software applications will also be executed separately as in scenario-I and scenario-II. Yet, instead of running solely a single task as in scenario-II, several tasks are executed per a software application. For instance, creating a new file, saving the newly created file, opening an already saved file, saving an already saved file in different name, browsing several webpages, opening several tabs, vising a secured webpages *("websites supporting https protocol")*. Visiting public or unsecured webpages *("webpages that don't support https protocol")*, sending and receiving emails *("with attachment and without attachments")*, running a video from the PC and from the chrome, uploading a file to Dropbox and OneDrive, downloading a file shared on Dropbox and OneDrive, sharing a file uploaded on Dropbox or OneDrive with a friend, running a program on Eclipse, compiling a program using Eclipse, etc. Again, it should be highlighted here that no 2 software application were ran at the same time.

Scenario-IV: This scenario is the most complicated one amongst others. Here, scenario III is repeated. Yet, in this scenario more than one software application were executed at the same time. Several combinations were applied. In other words, different group of applications were executed in different timestamps and every time different operations were implemented as described in scenario-III.

The above mentioned scenarios were repeated 10 times in different timespans. Finally, a dataset contains 126,831 examples as shown in Table 2. The input attributes that were considered in the training dataset were depicted in Table 3. Such features were adopted due to the fact that they have already been confirmed to be effective in DF analysis (Cho and Rogers, 2011; Carrier, 2005; Nelson et al., 2016).

Table 2
Examples distribution in the gathered dataset.

Software Applications	Total Examples
PowerPoint	10,887
Excel	10,933
Access	10,230
Word	12,570
Photoshop	11,014
Acrobat Reader	10,423
Chrome	13,977
Eclipse	9803
Windows Movie Maker	8112
Weka	7927
OneDrive	10,646
Dropbox	10,309
Total	126,831

4.3. Diving the dataset into training and testing datasets

The majority of DM and ML models are actually vulnerable to overfitting. Meaning, as the error ratio during the training phase decreases, the model produces inaccurate results when applied to some unseen examples. To handle this challenge, the *"Hold-Out"* (Witten et al., 2011) validation strategy is utilized in this research, through splitting the gathered dataset into training and testing sub-sets. The set of instances located in every single dataset were chosen arbitrarily. Following some previous studies (Mohammad et al., 2013b; Mohammad et al., 2013c) the collected data is separated into 70% for training the model and 30% for evaluating *"testing"* it. The former one is employed for training the model, whereas the later one will remain hidden and finally it will be utilized for evaluating the produced model once the training phase ends. This way the evaluation metrics obtained from the

Please cite this article as: R. M. A. Mohammad, An Enhanced Multiclass Support Vector Machine Model and its Application to Classifying File Systems Affected by a Digital Crime, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.10.010

Table 3
Set of input features.

No	Description	Data type
1	Event Date	Date
2	Event Time	Time
3	User	Text
4	Computer Name	Text
5	Event ID	Numerical
6	Type	Text
7	C Time - File Creation	Time
8	A Time - File Altered	Time
9	M Time - MFT Changed	Time
10	R Time - File Read	Time
11	DOS File Permissions *(Read-only, Hidden, System,Archive, Temporary, Compressed, Offline, Encrypted)*	Text
12	Owner Id	Numerical
13	Allocated size of the file	Numerical
14	Real size of the file	Numerical
15	Flags *(Directory, Compressed, Hidden)*	Text
16	Filename length in characters	Numerical
17	Filename namespace *(POSIX Win32, DOS, Win32&DOS)*	Text
18	Object Id *(Unique Id assigned to file)*	Numerical
19	Birth Volume Id *(Volume ivhere file was created)*	Numerical
20	Birth Object Id (Original Object Id of file)	Numerical
21	Domain Id *(Domain in which object was created)*	Numerical
22	Access Control type *(Access Ailoived, AccessDenied, System Audit)*	Text

testing dataset provide fair assessment of the model's generalization ability.

4.4. Dataset discretization

Data discretization is a technique of placing values into categories to be able to decreasing the number of available states an attribute holds. The formed categories will be viewed as separated values. Additional information on the subject of discretization are available in (Witten et al., 2011). In the present study, several attributes needs to be discretized prior they provided to the suggested SVM model. Among others, we may mention *"Allocated size of the file, and Real Size of the File"*. Amongst the pre-processing concerns is that several attributes contain textual phrases. Nevertheless, usually DM and ML methods may merely handle numerical values. For addressing this challenge, a vocabulary bag has been created by utilizing *"word2vect tool"* (Product, 2013). Such a tool has a practical application of *"continuous bag-of-words and skip-gram architectures"* to get numeric vectors depiction of the key phrases. Such representations could be consequently employed in various DM and ML algorithms. To elaborate further, it designates an index for every word contained in the attribute. These indexes will after that be normalized as described in Section. Nonetheless, even though some attributes contain texts, the nuber of the possible texts are reasonably limited. For example, *"Flags, DOS File Permissions, Access Control type, and Filename namespace"*. For that reason every possible value can be given a numeric value begins by 1 . As an example, the available values of the *"Flag"* attribute will be transformed to: *"Directory = 1, Compressed = 2, and Hidden = 3"*.

4.5. Dataset cleansing

The dataset cleansing phase is aimed at managing missing values as well as outliers. The outcome of this phase is often a more reliable dataset and therefore better quality model. Two techniques can be employed for managing missing values these are:

- Permanently eliminating the instance contains missing value(s).
- Substitute missing value(s). In other words, instances that include missing values are not required to be eliminated; they might be substituted by alternative value.

Thanks to MATLAB (MathWorks, 2019) which facilitates applying the two possible options above. For instance, one can employ *"fillmissing"* (MathWorks, 2019) for replacing any missing value with an alternative value, or on the other hand make use of *"rmmissing"* (MathWorks, 2019) for completely removing any instance containing a missing value. Through this study, the researchers utilized the second solution. Nevertheless, outlier existence in the training dataset may alter and so mislead the training procedure for the DM and ML techniques leading to much longer training time, inaccurate model, and inevitably unreliable outcomes. In this study, a MATLAB embedded function is utilized for the purpose of handling the outliers and is called *"rmoutliers"* (MathWorks, 2019). Using this function we tend to make sure that the most indicative set of instances will be maintained for further processing. In contrast, the instances which may weaken the performance of the produced model will be pulled from the dataset, hence, got completely no part during creating the resulting model.

4.6. Dataset normalization

If the range of the attributes varies considerably that means one attribute may take precedence over other attributes. Consequently, suitable normalization technique can be employed to minimize outweighing attributes which contain wider ranges. As an example, attributes that contain time and date values would have a wider range if they were depicted as numerical values. Among the many frequently applied strategies is scaling attribute's values in a predetermined scope. For example $[-1 \ldots 1]$ or $[0 \ldots 1]$. The current research incorporates the built-in function namely *"rescale"* (MathWorks, 2019) to normalize the gathered dataset. It is good to draw the readers attention to the fact that the time and date related attributes represented as OS ticks.

4.7. Compared DM and ML approaches and their parameter settings

This section evaluates and compares the applicability of the proposed model with some well-known DM approaches these are as follows:

1. Neural Networks (ANN) and also frequently known as "Artificial Neural Network (ANN)" (Mohammad, 2018). This is a computer-based model that mimic the human's mind and his nerve system which usually includes a group of nodes where everyone signifies a processing unit or a neuron. Each node's outcome is determined by a variable referred to as interconnection weight or relation strength. Traditionally, the weights will be fine-tuned by means of an error modification rule known as delta rule or *"Widrow-Hoff learning rule"* (Widrow and Michael, 1990; Mohammad et al., 2016; Mohammad et al., 2016). The NN structure utilized in the current research is a feed-forward topology containing a single hidden layer that is often denoted as *"multi-layered perceptron"*. Domains which require multiple hidden layers are actually rarely experienced (Widrow and Michael, 1990; Mohammad and AbuMansour, 2017; Kaufmann et al., 2011). Selecting how many hidden layer to use is actually a small piece of the dilemma when building a NN model. How many neurons inside the hidden layers needs to be identified properly. An inadequate number of hidden neurons can result in what is normally known as "under-fitting". However, employing lots of neurons can lead to producing an "overfitting" model (Widrow and Michael, 1990; Mohammad et al., 2013b; Mohammad et al., 2013c). Sigmoid activation is employed when constructing the model, simply because it is the most frequently used one among others due to non-

Please cite this article as: R. M. A. Mohammad, An Enhanced Multiclass Support Vector Machine Model and its Application to Classifying File Systems Affected by a Digital Crime, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.10.010

linearity as well as derivation easiness (Widrow and Michael, 1990; Witten et al., 2011). The learning rate is actually a crucial variable need for building all NN models. Whenever the learning rate is very little, the NN will usually tend to take long period of time in order to build the final model which could be vulnerable to *"overfitting"*. Conversely, in case the learning rate is large, the NN will learn more quickly but with the chance of producing a model that is criticized of being imperfect. There are other variables do significant part during building NN models, particularly: *"Momentum Factor"* and as well *"Epochs Size"*.

For the purpose of this research, a MATLAB function has been created for finding the optimal hyper-parameters of the NN model. The function allows the user to select how many possible hidden layers to create, how many possible hidden neurons in every layer, as well as the maximum number of epochs. Then, the function applies the 10-fold cross validation of each possible hyper-parameters combination. Finally, the best group of parameters are produced and are used for building the final model.

2. Random Forest (RF) (Witten et al., 2011). This method makes a forest and then make it random in some way. Each *"forest"* is regarded as an ensemble of *"Decision Trees"* which usually trained using *"bagging"* technique. RF may be applied for either regression or classification cases, and these make up the most of recent DM and ML systems.

The total number of trees to utilize when building the RF model is adjusted in the range from 100 until 1000, however, the step size was set to 100. Keep in mind that more trees does not mean the better outcomes should made. To illustrate, at a particular stage the advantage in the model's performance by learning additional trees might be less than the cost of calculation time for learning such trees (Nitze et al., 2012).

3. Bayesian Network (BN) (Friedman et al., 1997). BN is an easy, but at the same time powerful DM technique and so commonly used DM technique. BN acts as a probabilistic classifier that produces classification decisions by using *"Maximum A Posteriori"* decision procedure within a Bayesian mode.

Several varied evaluation strategies could be utilized for building a BN model, these are *"SimpleEstimator (BN-SIM)"* as well as *"BMAEstimator (BN-BMA)"* (Witten et al., 2011). Such evaluation strategies could be used for producing a BN model (Witten et al., 2011). The former strategy is utilized for calculating the conditional probability table as soon as the structure is learnt. Yet, the later one calculates the conditional probability table by means of *"Bayes Model Averaging (BMA)"*.

Selecting all these methods is mainly because they employ varied approaches when it comes to generating classifiers and they are widely used towards constructing robust classification models (Aburrous et al., 2010; Mohammad et al., 2012; Mohammad et al., 2013a).

4.8. Evaluation criteria

Normally, four evaluation criteria are considered for assessing the performance on any DM model namely: *"True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN)"*. In this research, in order to be consistent with previous research studies these four classification criteria will be utilized. Whereby TP denotes how many positive instances accurately labeled as such, FN signifies how many positive instances inaccurately labeled as negative. On the other hand, FP refers to the amount of negative instances wrongly categorized as positive, whereas, TN represents the amount of negative instances accurately labeled as so. Many different evaluation metrics are often resulting from these criteria including Precision *(P)*, and Recall *(R)*.

1. *P (*"*also knows as positive predictive Value*"*)*, it denotes the portion of relevant examples amongst the retrieved ones. Normally *P* is calculated as shown in equation (12).

$$P = \frac{TP}{TP + FP} \tag{12}$$

2. *R (*"*also known as sensitivity*"*)* represents the portion of relevant examples that are retrieved over the total number of relevant examples and it can be calculated as shown in equation (13).

$$R = \frac{TP}{TP + Fn} \tag{13}$$

Generally speaking, *P* is seen as a way of measuring exactness or quality, however *R* is the way of measuring completeness or quantity.

3. Another evaluation criterion that is considered in this research is the *"Harmonic Mean"* also known as F1-Score. This evaluation criterion weights R and P evenly, where a proper model should increase both of R and P at the same time. Hence, a model that reasonably functioning good on both would be preferred over good results on one and insufficient results on the other. Usually, *F1-Score* is calculated as in equation (14).

$$F1 = \frac{2\ P\ R}{P + R} \tag{14}$$

4. One more evaluation criterion employed in this research is actually the most commonly used one in all previous studies, that is the *"Accuracy"*, and it is usually calculated as shown in equation (15). Accuracy signifies the actual ratio of accurately labeled instances with regards to the overall number of examples inside the testing datasets.

$$Accuracy = \frac{TN + TP}{TN + FN + TP + FP} \tag{15}$$

5. Discussing the obtained results

This section compares the ability of the suggested model against different DM methods *(Section 4.7)* in determining what software application *(from the group of applications listed in* Table 2*)* accessed which system file(s). Further, it will shed lights on several important results attained from the evaluation tests. The assessment is going to be mainly according to the evaluation criteria which were reviewed through Section 4.8. The overall results are depicted in Table 4 and Fig. 4.

The results showed that the *EMSVM* outperforms all of considered DM and ML techniques. For instance, in term of the obtained accuracies, the *EMSVM* beats NN, RF, BN-SIM, and BN-BMA by 1.31%, 1.31%, 3.14%, 2.94% respectively. Yet, in terms of F1-score that considers R and P and weights them evenly, the *EMSVM* beats all other counted DM and ML in the sense that the F1-score obtained from the *EMSVM* exceeds that obtained from NN, RF, BN-SIM, and BN-BMA by 1.31%, 1.41%, 3.47%, and 3.26% respectively. These results prove that the proposed *EMSVM* method is able to build a model that is talented enough to enhance the classification performance. One reason behind obtaining such results is due to the robust method used for building the classification model.

The recommended parameter settings technique might be another reason for the sound results obtained from the *EMSVM*.

Please cite this article as: R. M. A. Mohammad, An Enhanced Multiclass Support Vector Machine Model and its Application to Classifying File Systems Affected by a Digital Crime, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.10.010

Table 4
Obtained experimental results.

Algorithm	Accuracy	Precision	Recall	F1-Measure
NN	91.41%	90.35%	91.68%	91.07%
RF	91.41%	91.68%	90.35%	90.97%
BN-SIM	89.58%	90.25%	87.58%	88.91%
BN-BMA	89.78%	89.84%	88.50%	89.12%
EMSVM	92.72%	91.68%	93.02%	92.38%

That is beside the robust method employed by the *EMSVM* when dealing with multi-class classification domains.

Interestingly, although the results obtained from the BN are somehow acceptable, the BN algorithms produced the worst classification accuracy and F1-score. That can be attributed due to that fact that BN does not depict attributes that are correlated. In other words, BN can effectively perform well in one direction relationship but not in bi or multi directional relationships. This readable result is actually confirms that the collected dataset contains some attributes that in fact linked to each other. That in fact explains why the produced BN model was not able to explain some significant details.

Remarkably, NN produced the second best classification performance and that indeed confirms the powerful capabilities of the NN classification models. One more observation attained from the experimental results is that the *EMSVM* produced the highest precision ration at 93.02% knowing that this criterion is essential in measuring the model's exactness. Not only this but also, the *EMSVM* surpasses other algorithms when comparing the completeness capability *(Recall ratio)* with others. All in all, the generalization ability of the proposed *EMSVM* is undoubtedly adequate as compared to several DM and ML algorithms.

The obtained results from the considered DM, ML algorithms and the *EMSVM* were acceptable. Such results indeed indicate the appropriateness of the DM and ML techniques in general and the *EMSVM* in particular in analysing incriminating digital evidence by inspecting the historical activities of file systems to realize if a malicious application program manipulated them.

One of the recommendations that could be concluded from the experiments is to find a way for dealing with the highly correlated features. To elaborate, when running more than one software applications they might access some shared system files and it could be difficult to decide which application really accessed a particular file system at a specific period of time. Running two or more application simultaneously is one reason for producing a highly correlated dataset. Another reason is the background applications that might access the same system files although if just one software application is perform. Meaning that, running a single application is not really mean that one application is executed but there are some hidden applications that are working in the background.

In general, DF investigation process is frequently broken into three essential stages these are: gathering evidence and proofs, analyzing gathered data, and finally the obtained results will be scientifically documented, discussed, and presented. In the first phase, the investigators collect as much as viable of useful data associated with a particular computer criminal offense. Yet, data analysis is going to evaluate these gathered data to be able to develop probably valuable information *(facts related to a malevolent activity)*. Such a stage is actually the main topic of the current research. Without correct assessment, the collected data could be meaningless or more worse the investigation might have no benefits. Lastly, the attained result(s) will be offered to help the law enforcement officials for taking the proper and the appropriate action(s). Nevertheless, the last phase is actually outside the scope of our research.

Just like traditional forensics investigation techniques, it is crucial to digital forensics investigators to rebuild the criminal offense scene since that could assist in discovering several invisible facts. Meaning to say, the investigators will need to build a chronological rehearsal of the actions affecting the file-system of the grabbed PC (s). That actually might assist in showing how the user accessed to the pc under investigation and what activities he did on it. Additionally, rebuilding events chronologies could help in realizing the list of software programs performed by a specific user, the system file(s) which had been loaded, modified, or removed during a particular time frame. Yet, in order to offer decisive proofs to the court, the process of reaching these proofs needs to be evidently confirmed to be consistent with acknowledged strategies and techniques. Nevertheless, recognizing which system files were loaded, modified, or possibly removed by which software application is definitely an important stage on the way to correctly rebuilding events chronology.

The experimental results confirmed that the *EMSVM* was able to identify the set of systems files affected by in incedint and link

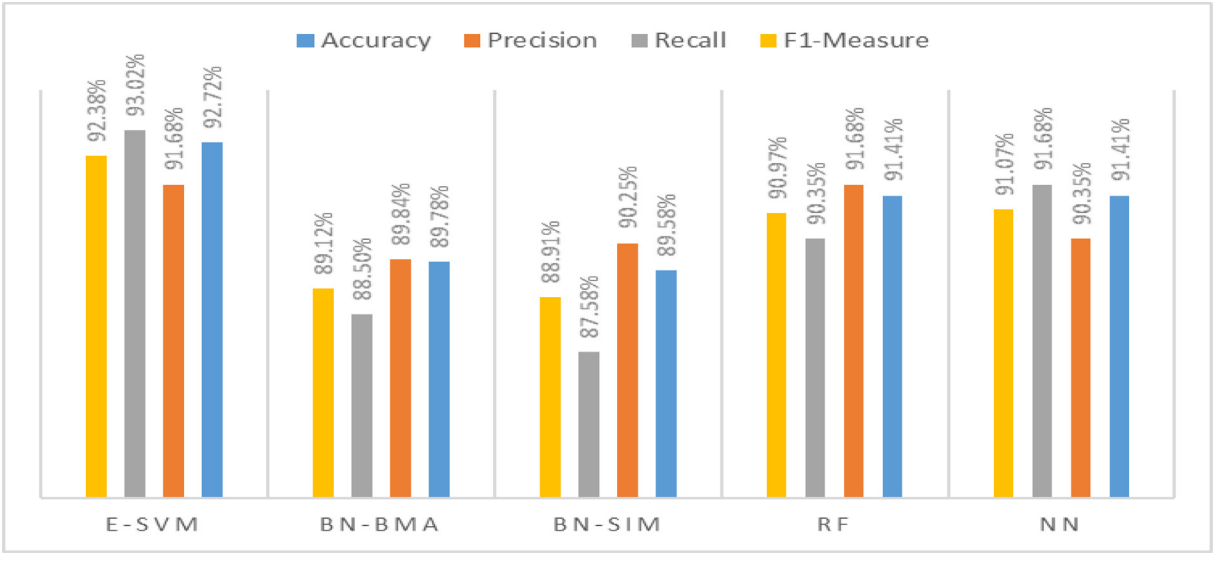

Fig. 4. Obtained Experimental Results.

Please cite this article as: R. M. A. Mohammad, An Enhanced Multiclass Support Vector Machine Model and its Application to Classifying File Systems Affected by a Digital Crime, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.10.010

them directly to the application program that changed their status. As a result, event reconstruction process would be more accurate and hence the weaknesses that were exploited in conducting the incedint will be addressed in order to avoid possible similar attacks in the future.

6. Conclusions

In this study, an enhanced multiclass SVM model was created that is essentially aimed to improve the classification abilities by firstly facilitating the process of selecting the most effective set of parameters when building a SVM model and secondly by enhancing the class assignment approach by supporting the multi-class classification domains. Such a model was used for analysing incriminating digital evidence by inspecting the historical actions that affecting the file systems status with the aim of recognizing if a malevolent application program has accessed, updated, or deleted any file system. The suggested model is talented enough to seek out the ideal parameters to be able to get the best classification performance. The performance of the proposed model has been compared to several other DM and ML techniques according to several evaluation metrics. A training dataset contains a number of fingerprints *("features or attributes")* related to file-system events *("file system metadata")*, PC registries, as well as system event logs has been gathered. Several scenarios were prepared in order to collect the training dataset examples. Every scenario tries to access file-system in various methods. The assessment results showed that the *EMSVM* surpasses all other considered DM and Ml algorithms in terms of different evaluation criteria namely: accuracy, F1-score, Recall, and Precision. Such results confirm the well structuring methodology used for building the *EMSVM*. In general, the generalization ability of the proposed *EMSVM* was adequate when compared to several DM and ML algorithms. Nonetheless, the results obtained from other DM and ML algorithms were also acceptable. Such results indeed indicate the appropriateness of the DM and ML techniques in general and the *EMSVM* in particular in analysing incriminating digital evidence by inspecting the historical activities of file systems to realize if a malicious application program manipulated them. Realizing the set of file system affected by a melovelent application would help in reconstruction of events related to a specific incedint and hence comprehend the weaknesses that might exist in the system which could be used for conducting similar attacks in the future. As a direct result, such weaknesses could be addressed and hence improving the system's security. As a possible future work, more investigation should be done to find a way to deal with the highly correlated features that could occur unintentionally when more than one application sharing the same file systems.

Ethical approval

This article does not contain any studies with human participants or animals performed by any of the authors.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Atif A., Ruighaver, A.B. "FIRESTORM: Exploring the Need for a Forensic Tool for Pattern Correlation in Windows NT Audit Logs. in: The 3rd Australian Information Warfare and Security Conference, 2002.

Aburrous, M., Hossain, M.A., Dahal, K., Thabtah, F., 2010. Intelligent phishing detection system for e-banking using fuzzy data mining. Expert Systems with Applications 37 (12), 7913–7921.

Basnet, R., Sung, A., H, L., Liu, Q., 2012. Feature Selection for Improved Phishing Detection. In: 25th International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems. Springer Berlin Heidelberg, pp. 252–261.

Bera, A., 2019. 83 Terrifying Cybercrime Statistics. safeatleast.

Bergstra, J., Bengio, Y., 2012. Random search for hyper-parameter optimization. J. Mach. Learn. Res. 13 (1), 281–305.

Carrier, B., 2005. In: File System Forensic Analysis. Addison Wesley Professional.

Carrier, B., Spafford, E., 2005. Automated Digital Evidence Target Definition Using Outlier Analysis and Existing Evidence. Digital Forensics Research. DFRWS, pp. 1–10.

Carvey, H., 2005. The windows registry as a forensic resource. Digital Investigat. 2 (3), 201–205.

Chabot, Y., Bertaux, A., Nicolle, C., 2014. A complete formalized knowledge representation model for advanced digital forensics timeline analysis. Digital Investigat. 11 (2), 95–105.

Cho, G.-S., Rogers, M.K., 2011. "Finding Forensic Information on Creating a Folder in $LogFile of NTFS," in *Third International ICST Conference, ICDF2C 2011*. Dublin, Ireland.

ClintCarr, M.R.C., Gunsch, G., 2002. An Examination of Digital Forensic. International Journal Digital Evidence 1 (3).

Cortes, C., Vapnik, V., 1995. Support-vector networks. Mach. Learn. 20 (3), 273–297.

DeVel, O. 2003. File Classification Using Sub-Sequence Kernels," in DigitalForensics Research.

Executive-Order-13402, "Executive Order 13402," 10, Online Available: http://www.gpo.gov/fdsys/pkg/FR-2006-05-15/pdf/06-4552.pdf May 2006 Accessed 14 March 2012.

Friedman, N. Geiger, D. Goldszmidt, M. 1997. Bayesian Network Classifiers," Machine Learning – Special issue on learning with probabilistic representations, vol. 29, no. 2-3, pp. 131–163.

Greene, D., Rosenblum, M., Devine, S., Wang, E., Bugnion, E., VMware, Inc., "VMware, Inc., February 1998. [Online]. Available: https://www.vmware.com/. (Accessed 1. 13.18).

Dongil, K., Kang, S., Cho, S., 2020. Expected margin–based pattern selection for support vector machines. Expert Systems with Applications.

Gonsalves, A.H., Thabtah, F., Mohammad, R.M., Singh, G., 2019. Prediction of Coronary Heart Disease using Machine Learning: An Experimental Analysis. The 2019 3rd International Conference on Deep Learning Technologies. ACM, pp. 51–56.

Henry, L., Timothy, P., Marilyn, M., 2007. Henry Lee's crime scene handbook. Elsevier.

Jin, Z., Chaorong, W., Chengguang H., Feng, W. 2014."Parameter optimization algorithm of SVM for fault classification in traction converter," in: The 26th Chinese Control and Decision Conference (2014 CCDC), Changsha, China.

Kaufmann, M. Han J. Pei, J. 2011. Data Mining: Concepts and Techniques, Elsevier, pp. 744.

Khemphila, A., Boonjing, V., 2011. "Heart Disease Classification Using Neural Network and Feature Selection," in *21st International Conference on Systems Engineering (ICSEng)*. Las Vegas, NV.

Lai, K.K., Yu, L., Zhou, L., Wang, S., 2008. Bio-Inspired Credit Risk Analysis: Computational Intelligence with Support Vector Machines. Springer.

Lee W. Xiang D. 2001. "Information-theoretic measures for anomaly detection," in IEEE Symposium on Security and Privacy, S&P 2001 2001 Oakland CA.

Ma, Y.F., Chen, F., 2015. A classification algorithm of moving military vehicle. Adv. Mater. Res. 989–994, 2043–2046.

MathWorks, "MathWorks Documentation," MathWorks, 1994-2019. [Online]. Available: https://www.mathworks.com/help/matlab/preprocessing-data.html. (Accessed 10 Jan 2018).

Mohammad, R.M., 2016. In: An, ensemble self-structuring neural network approach to solving classification problems with virtual concept drift and its application to phishing websites. University of Huddersfield, United Kingdom.

Mohammad, R.M., Alqahtani, M., 2019. A comparison of machine learning techniques for file system forensics analysis. J. Inform. Secur. Appl. 46 (1), 53–61.

Mohammad, R.M., AbuMansour, H.Y. 2017. An intelligent model for trustworthiness evaluation in semantic web applications, in: 8th International Conference on Information and Communication Systems (ICICS) Irbid Jordan.

Mohammad, R.M., Thabtah, F., McCluskey, L. 2016. An Improved Self-Structuring Neural Network. In: Pacific Asia Knowledge Discovery and Data Mining Conference (PAKDD) Auckland 2016.

Mohammad M., R., 2018. A Neural Network based Digital Forensics Classification. 2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA). IEEE, pp. 1–7.

Zahra Nazari Dongshik Kang Density Based Support Vector Machines for Classification ijarai 4 4 10.14569/issn.2165-4069 10.14569/IJARAI.2015.040411 http://ijarai.thesai.org/ http://thesai.org/Publications/ViewPaper?Volume=4&Issue=4&Code=ijarai&SerialNo=11.

Mohammad, R.M., Thabtah, F., McCluskey, L., 2012. An assessment of features related to phishing websites using an automated technique. International Conference for Internet Technology And Secured Transactions. IEEE.

Mohammad, R.M., Thabtah, F., McCluskey, L., 2013a. Intelligent Rule based Phishing Websites Classification. IET Information Security 1, 153–160. https://doi.org/10.1049/iet-ifs.2013.0202.

Please cite this article as: R. M. A. Mohammad, An Enhanced Multiclass Support Vector Machine Model and its Application to Classifying File Systems Affected by a Digital Crime, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.10.010

Mohammad, R.M., Thabtah, F., McCluskey, L., 2013b. Predicting phishing websites based on self-structuring neural network. Neural Computing and Applications, 443–458. https://doi.org/10.1007/s00521-013-1490-z.

Mohammad, R., Thabtah, F., McCluskey, L., 2013c. Predicting Phishing Websites using Neural Network trained with Back-Propagation. In: ICAI. World Congress in Computer Science, Computer Engineering, and Applied Computing, pp. 682–686.

Mohammad, R.M., Thabtah, F., McCluskey, L., 2015. Tutorial and critical analysis of phishing websites methods. Computer Science Review 17 (1), 1–24. https://doi.org/10.1016/j.cosrev.2015.04.001.

Mohammad, R., Thabtah, F., McCluskey, L., 2016. A dynamic self-structuring neural network model to combat phishing. In: International Joint Conference on Neural Networks (IJCNN). IEEE, pp. 4221–4226.

Nelson, B., Phillips, A., Steuart, C., 2016. Guide to Computer Forensics and Investigations. Course Technology Inc.

Nitze, I. Schulthess, U. Asche, H. 2012. "Comparison of Machine Learning Algorithms Random Forest, Artificial Neural Network and Support Vector Machine to Maximum Likelihood for Supervised Crop Type Classification," In: Proceedings of the 4th GEOBIA, Rio de Janeiro.

Pollitt, M. 1995. "Computer Forensics: An Approach to Evidence in Cyberspace," in The National Information Systems Security, Baltimore.

G. Product, "Word2vec," Google Code, 30 Jul 2013. [Online]. Available: https://code.google.com/archive/p/word2vec/. (Accessed 12 Mar 2017).

Rocha A., Goldenstein, S.K. 204. "Multiclass from Binary: Expanding One-Versus-All, One-Versus-One and ECOC-Based Approaches, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, vol. 25, no. 2, pp. 289–302.

Rynearson, J. M. 2002. Evidence and crime scene reconstruction, National Crime Investigation and Training.

Schatz B., Mohay G., Clark A., 2004. "Rich event representation for computer forensics," in The Fifth Asia-Pacific Industrial Engineering and Management Systems.

Palmer, G., 2001. A Road Map for Digital Forensic Research. Digital Forensic Research Workshop (DFRWS), 1–49.

Powell, M., 2019. 11 Eye Opening Cyber Security Statistics for 2019. CPO Magazine.

Qiu, H., Qiu, M., Lu, Z., 2020. Selective encryption on ECG data in body sensor network based on supervised machine learning. Information Fusion, 59–67. https://doi.org/10.1016/j.inffus.2019.07.012.

Tzu-Liang, T., Aleti, K.R., Hu, Z., Kwon, Y.J., 2015. E-quality control: a support vector machines approach. J. Comput. Design Eng. 3 (1), 91–101.

Vel, O.D., Anderson, A., Corney, M., Mohay, G., 2001. Mining e-mail content for author identification forensics. ACM SIGMOD 30 (4), 55–64.

Witten, I.H. Frank, E. Mark A.H. 2011. ThirdData mining: practical machine learning tools and techniques with Java implementations. Morgan Kaufmann.

Widrow, B., Michael, L., 1990. 30 years of adaptive neural networks. IEEE press, 1415–1442.

Wu, X., Son, C., Zou, T., Li, L., Liu, H., 2020. SVM-based image partitioning for vision recognition of AGV guide paths under complex illumination conditions. Robotics and Computer-Integrated Manufacturing. https://doi.org/10.1016/j.rcim.2019.101856.

Zhu, Y. Gladyshev P. James J., 2009. Using shellbag information to reconstruct user activities, in: The Ninth Annual DFRWS Conference.

NirSoft. 2001. (accessed 20 March 2018).

Sleuth Kit. 2003. (accessed 5 April 2018).

SANS DFIR. 2008. (accessed 18 January 2018).

Please cite this article as: R. M. A. Mohammad, An Enhanced Multiclass Support Vector Machine Model and its Application to Classifying File Systems Affected by a Digital Crime, Journal of King Saud University – Computer and Information Sciences, https://doi.org/10.1016/j.jksuci.2019.10.010