METHOD AND SYSTEM FOR TRAINING MACHINE LEARNING ENGINE AND RELATED DEVICE

[0001] The present application claims priority to Chinese Patent Application No.201810638258.2, titled "METHOD AND SYSTEM FOR TRAINING MACHINE LEARNING ENGINE AND RELATED DEVICE", filed on June 20, 2018 with the China National Intellectual Property Administration, which is incorporated herein by reference in its entirety.

FIELD

[0002] The present disclosure relates to the field of machine learning, and in particular to a method and a system for training a machine learning engine, a computer-readable storage medium and a device for training a machine learning engine.

BACKGROUND

[0003] Currently, the technology of big data and machine learning is a research hotspot in industry and academia. The technology of machine learning and big data is increasingly applied to information security, achieving good results. Malware detection technology based on machine learning has made a great progress, and a large number of companies, such as Cylance, providing malware detection products based on machine learning have emerged in the industry. The emerging companies are favored by VC and have achieved many good results.

[0004] In the conventional machine learning engines, information entropy of a software is determined as static features for the machine learning. Generally, the information entropy is extracted from main features of an executable file, and the extracted information entropy is used as static features to be inputted to the machine learning engine. However, the operation of extracting information entropy from the main features of the executable file is only performed on an import table, an export table and section information of the executable file, the import table, the export table and the section information are general features, thus the extracted sample features have coarse granularity. The machine learning based on the sample features according to the conventional technology will result in a poor effect in training the

machine learning engine and a low recognition rate.

[0005] Therefore, how to stably extract the static features of the executable file to make the detection model generated by the machine learning engine have a recognition rate with strong robust is a problem currently required to be solved by those skilled in the art.

SUMMARY

[0006] A method and a system for training a machine learning engine, a computer-readable storage medium and a device for training a machine learning engine are provided according to the present disclosure, to stably extract static features of an executable file to make a detection model generated by a machine learning engine have a recognition rate with strong robust.

[0007] To solve the above technical problems, a method for training a machine learning engine is provided according to the present disclosure. The method includes:

obtaining an executable file, and dividing the executable file into multiple to-be-extracted parts based on a data directory of the executable file;

extracting information entropy features of all the to-be-extracted parts, and performing data transformation on the information entropy features to obtain static features; and

training a machine learning engine using the static features.

[0008] In an embodiment, the dividing the executable file into multiple to-be-extracted parts based on a data directory of the executable file includes:

dividing the executable file into an executable file header part, an executable file section part, an executable file Overlay data part, and an executable file integral part based on the data directory of the executable file.

[0009] In an embodiment, the extracting information entropy features of all the to-be-extracted parts and performing data transformation on the information entropy features to obtain static features includes:

extracting the information entropy features of all the to-be-extracted parts, where the information entropy features are binary data;

calculating a proportion of each of characters in the binary data corresponding to the information entropy features to all the characters in the binary data, and obtaining an entropy

flow of the information entropy features based on the proportion; and

performing a DCT transform or a DWT transform on the entropy flow to obtain the static features.

[0010] In an embodiment, the calculating a proportion of each of characters in the binary data corresponding to the information entropy features to all the characters in the binary data and obtaining an entropy flow of the information entropy features based on the proportion includes:

dividing the binary data corresponding to the information entropy features into a plurality of byte blocks, where each of the byte blocks includes 256 bytes;

for each of the byte blocks, calculating a byte block proportion p of each of characters in the byte block to all characters in the byte block, and calculating a entropy flow component S for each of bytes in the byte block by using the following equation:

$$S = \left| p \cdot \log(p) \right|; \text{ and}$$

accumulating all the entropy flow components to obtain the entropy flow.

[0011] In an embodiment, after obtaining the executable file, the method further includes:

determining a file format of the executable file based on general format information of the executable file; and selecting a feature extraction strategy based on the file format.

[0012] The extracting information entropy features of all the to-be-extracted parts includes: extracting the information entropy features of all the to-be-extracted parts based on the feature extraction strategy.

[0013] In an embodiment, the determining a file format of the executable file based on general format information of the executable file includes:

determining values of .Net MetaData Directory RVA and .Net MetaData Directory Size in the data directory of the executable file; and

determining, if the values of the .Net MetaData Directory RVA and the .Net MetaData Directory Size are not empty, that the file format of the executable file is .Net format.

[0014] In an embodiment, the dividing the executable file into multiple to-be-extracted

parts based on a data directory of the executable file includes:

dividing the executable file into multiple to-be-extracted parts based on a section array of the executable file.

[0015] In an embodiment, after training a machine learning engine using the static features, the method further includes:

extracting, in a case that a suspicious software is detected, a target executable file of the suspicious software; and

inputting the target executable file into the trained machine learning engine to determine, by the trained machine learning engine, whether the target executable file is an executable file of a malware.

[0016] In an embodiment, the method further includes:

reporting, if it is determined by the trained machine learning engine that the target executable file is the executable file of the malware, software information of the suspicious software to perform manual detection on the suspicious software;

receiving a manual detection result, and determining whether the current determination by the trained machine learning engine is wrong based on the manual detection result; and

training, if the current determination by the trained machine learning engine is wrong, the machine learning engine using the target executable file as a new negative training sample.

[0017] A system for training a machine learning engine is further provided according to the present disclosure. The system includes a file division module, a static feature extraction module and a training module.

[0018] The file division module is configured to obtain an executable file and divide the executable file into multiple to-be-extracted parts based on a data directory of the executable file.

[0019] The static feature extraction module is configured to extract information entropy features of all the to-be-extracted parts and perform data transformation on the information entropy features to obtain static features.

[0020] The training module is configured to train a machine learning engine using the static

features.

[0021] In an embodiment, the file division module includes a file obtaining unit and a division unit.

[0022] The file obtaining unit is configured to obtain the executable file.

[0023] The division unit is configured to divide the executable file into an executable file header part, an executable file section part, an executable file Overlay data part, and an executable file integral part based on the data directory of the executable file.

[0024] In an embodiment, the static feature extraction module includes an information entropy feature extraction unit, an entropy flow calculation unit and a data transformation unit.

[0025] The information entropy feature extraction unit is configured to extract the information entropy features of all the to-be-extracted pars. The information entropy feature are binary data.

[0026] The entropy flow calculation unit is configured to calculate a proportion of each of characters in the binary data corresponding to the information entropy features to all the characters in the binary data and obtain an entropy flow of the information entropy features based on the proportion.

[0027] The data transformation unit is configured to perform a DCT transform or a DWT transform on the entropy flow to obtain the static features.

[0028] In an embodiment, the entropy flow calculation unit includes a division sub-unit, an entropy flow component calculation sub-unit and an entropy flow component accumulation unit.

[0029] The division sub-unit is configured to divide the binary data corresponding to the information entropy features into multiple byte blocks, where each of the byte blocks includes 256 bytes.

[0030] The entropy flow component calculation sub-unit is configured to, for each of the byte blocks, calculate a byte block proportion p of each of characters in the byte block to all characters in the byte block, and calculate a entropy flow component S for each of bytes in

the byte block by using the following equation:

$$S = \left| p \cdot \log(p) \right|.$$

[0031] The entropy flow component accumulation unit is configured to accumulate all the entropy flow components to obtain the entropy flow.

[0032] In an embodiment, the system further includes an extraction strategy determination module.

[0033] The extraction strategy determination module is configured to determine a file format of the executable file based on general format information of the executable file, and select a feature extraction strategy based on the file format.

[0034] The static feature extraction module is configured to extract the information entropy features of all the to-be-extracted parts based on the feature extraction strategy, and perform data transformation on the information entropy features to obtain the static features.

[0035] In an embodiment, the extraction strategy determination module includes a file format determination unit and a strategy selection unit.

[0036] The file format determining unit is configured to determine values of .Net MetaData Directory RVA and .Net MetaData Directory Size in the data directory of the executable file, and determine, if the values of the .Net MetaData Directory RVA and the .Net MetaData Directory Size are not empty, that the file format of the executable file is .Net format.

[0037] The strategy selection unit is configured to select the feature extraction strategy based on the file format.

[0038] In an embodiment, the file division module is configured to obtain the executable file and divide the executable file into the multiple to-be-extracted parts based on a section array of the executable file.

[0039] In an embodiment, the system further includes an extraction module and a determination module.

[0040] The extraction module is configured to extract, in a case that a suspicious software is detected, a target executable file of the suspicious software.

[0041] The determination module is configured to input the target executable file into the

trained machine learning engine to determine, by the trained machine learning engine, whether the target executable file is an executable file of a malware.

[0042] In an embodiment, the system further includes a reporting module and a secondary training module.

[0043] The reporting module is configured to report, if it is determined by the trained machine learning engine that the target executable file is the executable file of the malware, software information of the suspicious software to perform manual detection on the suspicious software.

[0044] The secondary training module is configured to receive a manual detection result, determine whether the current determination by the trained machine learning engine is wrong based on the manual detection result, and train, if the current determination by the trained machine learning engine is wrong, the machine learning engine using the target executable file as a new negative training sample.

[0045] A computer-readable storage medium storing a computer program is further provided according to the present disclosure. The computer program, when executed, performs the method for training a machine learning engine.

[0046] A device for training a machine learning engine is further provided according to the present disclosure. The device includes a memory and a processor. The memory stores a computer program. The processor is configured to execute the computer program to perform the method for training a machine learning engine.

[0047] A method for training a machine learning engine is provided according to the present disclosure. The method includes: obtaining an executable file and dividing the executable file into multiple to-be-extracted parts based on a data directory of the executable file; extracting information entropy features of all the to-be-extracted parts and performing data transformation on the information entropy features to obtain static features; and training a machine learning engine using the static features.

[0048] In the present disclosure, the executable file is divided into multiple to-be-extracted parts, information entropy features of all the to-be-extracted parts are extracted, an transformation is performed on the extracted information entropy features to obtain static features which are required in training the machine learning engine to train the machine

learning engine. Compared with the method according to the conventional technology in which only the information entropy of the main features of the executable file are extracted, with the method in the present disclosure, the executable file is divided into multiple to-be-extracted parts based on the data directory and information entropy features of all the to-be-extracted parts are extracted. According to the conventional technology, since the information entropy features are extracted from the main features which are common features, the obtained static features have a coarse granularity. According to the present disclosure, the information entropy features are extracted from the entire executable file by dividing the executable file into multiple to-be-extracted parts and extracting the information entropy features of all the to-be-extracted parts, thus it can be ensured that the obtained static features have a fine granularity. With the method according to the present disclosure, the static features of the executable file can be stably extracted, so that the detection model generated by the machine learning engine has a recognition rate with strong robust. According to the present disclosure, a system for training a machine learning engine, a computer-readable storage medium and a device for training a machine learning engine are further provided, which have the above beneficial effects and are not repeated herein.

BRIEF DESCRIPTION OF THE DRAWINGS

[0049] In order to illustrate the embodiments of the present disclosure or technical solutions in the conventional technology more clearly, drawings to be used in the description of the embodiments or the conventional technology are introduced simply hereinafter. It is apparent that the drawings described below only illustrate some embodiments of the present disclosure. For those skilled in the art, other drawings may be obtained according to these drawings without any creative work.

[0050] Figure 1 is a flow chart of a method for training a machine learning engine according to an embodiment of the present disclosure;

[0051] Figure 2 is a flow chart of extracting information entropy features and performing transformation on the extracted information entropy features to obtain static features in a method for training a machine learning engine according to an embodiment of the present disclosure;

[0052] Figure 3 is a flow chart of a method for training a machine learning engine

according to another embodiment of the present disclosure;

[0053] Figure 4 is a flow chart of a method for training a machine learning engine according to another embodiment of the present disclosure; and

[0054] Figure 5 is a schematic structural diagram of a system for training a machine learning engine according to an embodiment of the present disclosure.

DETAILED DESCRIPTION

[0055] In order to make the objectives, technical solutions and advantages of the embodiments of the present disclosure clearer, technical solutions according to the embodiments of the present disclosure are described clearly and completely hereinafter in conjunction with the drawings used in the embodiments of the present disclosure. It is apparent that the described embodiments are only some embodiments of the present disclosure rather than all embodiments. All other embodiments obtained by those skilled in the art based on the embodiments of the present disclosure without any creative work fall within the protection scope of the present disclosure.

[0056] Reference is made to Figure 1, which is a flow chart of a method for training a machine learning engine according to an embodiment of the present disclosure.

[0057] The method includes the following steps S101 to S103.

[0058] In step S101, an executable file is obtained, and the executable file is divided into multiple to-be-extracted parts based on a data directory of the executable file.

[0059] In the embodiment, a method for training a machine learning engine is provided. Samples required for training the machine learning engine are static features of the executable file. The executable file is a file which may be loaded and executed by an operating system. In different operating system environments, the executable file is in different formats. For example, the executable file in a Windows system is a PE (portable executable) file, and the executable file in a system, such as Linux, Mac, and Android, has a file format corresponding to the system. The system where the executable file is executed is not limited here. According to the present disclosure, operations can be performed on the executable files in multiple systems.

[0060] In this step, the executable file is divided into multiple to-be-extracted parts based

on the data directory of the executable file. Specifically, the executable file may be divided based on a section array in the structure of the executable file, and may be divided based on other data directories in the structure of the executable file, which is not limited herein. For example, the executable file may be divided into an executable file header part, an executable file section part, an executable file Overlay data part, and an executable file integral part based on the data directory of the executable file . In an embodiment, after general format information of the executable file is determined, the executable file may be divided into the to-be-extracted parts based on some parts or some structures of the general format information of the executable file.

[0061] In this step, the executable file is divided into multiple to-be-extracted parts based on the data directory. For each of the to-be-extracted parts, the information entropy features may be extracted with a specific feature extraction method, achieving information entropy features with a fine granularity, thereby increasing the stability of the static features.

[0062] In step S102, information entropy features of all the to-be-extracted parts are extracted, and data transformation is performed on the information entropy features to obtain static features.

[0063] In this step, based on that the executable file has been divided into multiple to-be-extracted parts in step S101, the information entropy features of all the to-be-extracted parts are extracted. The information entropy is an occurrence probability of a kind of information, and may be compared with the thermodynamic entropy. The information entropy features in the embodiment are occurrence probabilities of all kinds of information in the to-be-extracted parts. Generally, information in a computer is in the form of characters. The information entropy features in this step may be considered as feature values of the occurrence probabilities of the characters in the executable file.

[0064] It should be understood that, the samples used for training the machine learning engine are static features of the executable files. In this step, data transformation is performed on the information entropy features to obtain the static features, so that the machine learning engine is trained in a next step.

[0065] In step S103, the machine learning engine is trained using the static features.

[0066] In the embodiment, the executable file is divided into multiple to-be-extracted parts,

information entropy features of all the to-be-extracted parts are extracted, and transformation is performed on the extracted information entropy features to obtain static features which are required in training the machine learning engine to train the machine learning engine. Compared with the method according to the conventional technology in which only the information entropy of the main features of the executable file are extracted, with the method in the embodiment, the executable file is divided into multiple to-be-extracted parts based on the data directory and information entropy features of all the to-be-extracted parts are extracted. According to the conventional technology, since the information entropy features are extracted from the main features which are common features, the obtained static features have a coarse granularity. According to the embodiment, the information entropy features are extracted from the entire executable file by dividing the executable file into multiple to-be-extracted parts and extracting the information entropy features of all the to-be-extracted parts, thus it can be ensured that the obtained static features have a fine granularity. With the method according to the embodiment, the static features of the executable file can be stably extracted, so that the detection model generated by the machine learning engine has a recognition rate with strong robust.

[0067] Reference is made to Figure 2, which is a flow chart of extracting information entropy features and performing transformation on the extracted information entropy features to obtain static features in a method for training a machine learning engine according to an embodiment of the present disclosure.

[0068] The extracting information entropy features and performing transformation on the extracted information entropy features to obtain static features includes but is not limited to the following steps.

[0069] In step S201, the information entropy features of all the to-be-extracted parts are extracted. The information entropy features are binary data.

[0070] In step S202, a proportion of each of characters in the binary data corresponding to the information entropy features to all the characters in the binary data is calculated, and an entropy flow of the information entropy features is obtained based on the proportion.

[0071] In the method according to the present disclosure, the information entropy may be compared with thermodynamic entropy, and the transmission of net entropy via information flow is called as entropy flow.

[0072] In this step, the entropy flow is calculated based on the overall information entropy features which are binary data. In an embodiment, the binary data may be divided into multiple blocks for calculating the entropy flow. For example, the binary data corresponding to the information entropy features is divided into multiple byte blocks, where each of the byte blocks includes 256 bytes. For each of the byte blocks, a byte block proportion p of each of characters in the byte block to all characters in the byte block is calculated, and an entropy flow component S for each of bytes in the byte block is calculated by using the equation:

$$S = \left| p \cdot \log(p) \right|$$

All the entropy flow components are accumulated to obtain the entropy flow.

[0073] In step S203, a DCT transform or a DWT transform is performed on the entropy flow to obtain the static features.

[0074] The DCT (Discrete Cosine Transform) is a transform related to Fourier transform. The DWT (Discrete Walsh Transform) is a non-sinusoidal orthogonal transform with a Walsh function as a basic function. The static features may be obtained by performing the DCT transform or the DWT transform on the entropy flow.

[0075] Reference is made to Figure 3, which is a flow chart of a method for training a machine learning engine according to another embodiment of the present disclosure.

[0076] The method includes the following steps S301 to S303.

[0077] In step S301, a file format of the executable file is determined based on general format information of the executable file, and a feature extraction strategy is selected based on the file format.

[0078] In this step, a subclass (such as, a .Net format, a Py2exe format, an auotit format and a Nsis format) of the executable file is determined based on the general format information of the executable file, and a feature extraction strategy (such as a strategy for determining which features are to be extracted) is determined based on the file format of the executable file.

[0079] The following are examples for determining the file format. In determining the file

format of the executable file, it may be determined whether values of ".Net MetaData Directory RVA" and ".Net MetaData Directory Size" in the data directory of the executable file are empty. If the values of the ".Net MetaData Directory RVA" and the ".Net MetaData Directory Size" in the data directory of the executable file are not empty, the executable file is in .Net format. Alternatively, it may be detected whether the executable file includes a reference of a library while python is run. For example, it is detected whether an import table of the executable file includes python*.dll format or other formats (whether including codes compiled with python or python codes) to determine whether the executable file is in Py2exe format. Executable files in different file formats may be determined in different ways, which are not limited here and may be configured by those skilled in the art according to actual application conditions.

[0080] In step S302, the static features of the executable file are extracted based on the feature extraction strategy.

[0081] Although the feature extraction strategies are different, the purpose for extracting the static features may be as follows: extracting certain structures and metadata from the executable file, reading and parsing the attributes of each of the structures and metadata according to a predefined structure, converting numeric data into a certain range, and performing operation, such as intercepting characters of a certain length or replacing special characters, on string data.

[0082] In step S303, the machine learning engine is trained using the static features.

[0083] With the technical solution according to the embodiment, static features having a fine granularity can be obtained to train the machine learning engine, improving the recognition rate of the machine learning engine. The operations performed in the embodiment may be combined with the operations performed in the first embodiment or the second embodiment, so that the information entropy features in the previous two embodiments are extracted based on the feature extraction strategy selected based on the file format according to the embodiment. That is, the information entropy features of the to-be-extracted parts are extracted based on the feature extraction strategy.

[0084] Reference is made to Figure 4, which is a flow chart of a method for training a machine learning engine according to another embodiment of the present disclosure.

[0085] In step S401, an executable file is obtained, and the executable file is divided into multiple to-be-extracted parts based on a data directory of the executable file.

[0086] In step S402, a file format of the executable file is determined based on general format information of the executable file, and a feature extraction strategy is selected based on the file format.

[0087] In step S403, information entropy features of all the to-be-extracted parts are extracted based on the feature extraction strategy. The information entropy features are binary data.

[0088] In step S404, the binary data corresponding to the information entropy features is divided into multiple byte blocks, where each of the byte blocks includes 256 bytes.

[0089] In step S405, for each of the byte blocks, a byte block proportion p of each of characters in the byte block to all characters in the byte block is calculated, and a entropy flow component S for each of bytes in the byte block is calculated by using the following equation:

$$S = \left| p \cdot \log(p) \right|$$

[0090] In step S406, all the entropy flow components are accumulated to obtain the entropy flow.

[0091] In step S407, a DCT transform or a DWT transform is performed on the entropy flow to obtain the static features.

[0092] In step S408, the machine learning engine is trained using the static features.

[0093] After training the machine learning engine using the static feature, the method may further include the following steps 1 to 4.

[0094] In step 1, in a case that a suspicious software is detected, a target executable file of the suspicious software is extracted.

[0095] In step 2, the target executable file is inputted into the trained machine learning engine to determine, by the trained machine learning engine, whether the target executable file is an executable file of a malware.

[0096] In step 3, if it is determined by the trained machine learning engine that the target executable file is the executable file of the malware, software information of the suspicious software is reported to perform manual detection on the suspicious software.

[0097] In step 4, a manual detection result is received, and it is determined whether the current determination by the trained machine learning engine is wrong based on the manual detection result. If the current determination by the trained machine learning engine is wrong, the machine learning engine is trained using the target executable file as a new negative training sample.

[0098] In the embodiment, after the machine learning engine is trained, the trained machine learning engine is used for detecting malware. Since the trained machine learning engine may have a low detection accuracy, in the embodiment, if the machine learning engine detects a malware, the malware is reported and a manual detection (that is, re-detection) is performed on the malware. If it is determined based on the manual detection that the suspicious software is not a malware, it indicates that the detection by the machine learning engine is wrong. The machine learning engine may be trained again using the target executable file of the suspicious software as a new negative training sample to improve the detection accuracy. Training samples include positive samples and negative samples. The positive samples correspond to executable files of malwares. The negative samples correspond to executable files of non-malwares.

[0099] Reference is made to Figure 5, which is a schematic structural diagram of a system for training a machine learning engine according to an embodiment of the present disclosure.

[0100] The system includes a file division module 100, a static feature extraction module 200 and a training module 300.

[0101] The file division module 100 is configured to obtain an executable file and divide the executable file into multiple to-be-extracted parts based on a data directory of the executable file.

[0102] The static feature extraction module 200 is configured to extract information entropy features of all the to-be-extracted parts and perform data transformation on the information entropy features to obtain static features.

[0103] The training module 300 is configured to train a machine learning engine using the

static features.

[0104] In an embodiment, the file division module 100 includes a file obtaining unit and a division unit.

[0105] The file obtaining unit is configured to obtain the executable file.

[0106] The division unit is configured to divide the executable file into an executable file header part, an executable file section part, an executable file Overlay data part, and an executable file integral part based on the data directory of the executable file.

[0107] In an embodiment, the static feature extraction module 200 includes an information entropy feature extraction unit, an entropy flow calculation unit and a data transformation unit.

[0108] The information entropy feature extraction unit is configured to extract the information entropy features of all the to-be-extracted parts. The information entropy features are binary data.

[0109] The entropy flow calculation unit is configured to calculate a proportion of each of characters in the binary data corresponding to the information entropy features to all the characters in the binary data and obtain an entropy flow of the information entropy features based on the proportion.

[0110] The data transformation unit is configured to perform a DCT transform or a DWT transform on the entropy flow to obtain the static features.

[0111] In an embodiment, the entropy flow calculation unit includes a division sub-unit, an entropy flow component calculation sub-unit and a entropy flow component accumulation unit.

[0112] The division sub-unit is configured to divide the binary data corresponding to the information entropy features into multiple byte blocks, where each of the byte blocks includes 256 bytes.

[0113] The entropy flow component calculation sub-unit is configured to, for each of the byte blocks, calculate a byte block proportion p of each of characters in the byte block to all characters in the byte block, and calculate a entropy flow component S for each of bytes in

the byte block by using the following equation:

$$S = \left| p \cdot \log(p) \right|.$$

[0114] The entropy flow component accumulation unit is configured to accumulate all the entropy flow components to obtain the entropy flow.

[0115] In an embodiment, the system further includes an extraction strategy determination module.

[0116] The extraction strategy determination module is configured to determine a file format of the executable file based on general format information of the executable file, and select a feature extraction strategy based on the file format.

[0117] The static feature extraction module is configured to extract the information entropy features of all the to-be-extracted parts based on the feature extraction strategy, and perform data transformation on the information entropy features to obtain the static features.

[0118] In an embodiment, the extraction strategy determination module includes a file format determination unit and a strategy selection unit.

[0119] The file format determining unit is configured to determine values of .Net MetaData Directory RVA and .Net MetaData Directory Size in the data directory of the executable file, and determine, if the values of the .Net MetaData Directory RVA and the .Net MetaData Directory Size are not empty, that the file format of the executable file is .Net format.

[0120] The strategy selection unit is configured to select the feature extraction strategy based on the file format.

[0121] In an embodiment, the file division module is configured to obtain the executable file and divide the executable file into the multiple to-be-extracted parts based on a section array of the executable file.

[0122] In an embodiment, the method further includes an extraction module and a determination module.

[0123] The extraction module is configured to extract, in a case that a suspicious software is detected, a target executable file of the suspicious software.

[0124] The determination module is configured to input the target executable file into the

trained machine learning engine to determine by the trained machine learning engine whether the target executable file is an executable file of a malware.

[0125] In an embodiment, the method further includes a reporting module and a secondary training module.

[0126] The reporting module is configured to report, if it is determined by the trained machine learning engine that the target executable file is the executable file of the malware, software information of the suspicious software to perform manual detection on the suspicious software.

[0127] The secondary training module is configured to receive a manual detection result, determine whether the current determination by the trained machine learning engine is wrong based on the manual detection result, and train, if the current determination by the trained machine learning engine is wrong, the machine learning engine using the target executable file as a new negative training sample.

[0128] Since the embodiments of the system correspond to the embodiments of the method, for the embodiments of the system, references may be made to the description of the embodiments of the method, which are not repeated herein.

[0129] A computer-readable storage medium storing a computer program is further provided according to the present disclosure. The computer program, when executed, perform the method according to the above embodiments. The storage medium may include mediums capable of storing program codes, such as a USB flash disk, a removable hard disk, a read-only memory (ROM), a random access memory (RAM), a magnetic disk or an optical disk.

[0130] A device for training a machine learning engine is further provided according to the present disclosure. The device may include a memory and a processor. The memory stores a computer program. The processor is configured to execute the computer program stored in the memory to perform the method according to the above embodiments. The device for training a machine learning engine may include various network interfaces, a power supply and other components.

[0131] The embodiments in this specification are described in a progressive way, each of which emphasizes the differences from others, and the same or similar parts among the

embodiments can be referred to each other. Since the system disclosed in the embodiments corresponds to the method therein, the description thereof is relatively simple, and for relevant matters references may be made to the description of the method. It should be noted that for those skilled in the art, many alterations and modifications may be made without departing from the principle of the present disclosure, and these alterations and modifications all fall within the scope of protection of the present disclosure.

[0132] It should be further noted that the relationship terminologies such as "first", "second" in the present disclosure are only used herein to distinguish one entity or operation from another, rather than to necessitate or imply that the actual relationship or order exists between the entities or operations. Furthermore, terms of "include", "comprise" or any other variants are intended to be non-exclusive. Therefore, a process, method, article or device including multiple components includes not only the components but also other components that are not enumerated, or also include the components inherent for the process, method, article or device. Unless expressively limited otherwise, the statement "comprising (including) one…" does not exclude the case that other similar components may exist in the process, method, article or device.

CLAIMS

1. A method for training a machine learning engine, comprising:

obtaining an executable file, and dividing the executable file into a plurality of to-be-extracted parts based on a data directory of the executable file;

extracting information entropy features of all the to-be-extracted parts, and performing data transformation on the information entropy features to obtain static features; and

training a machine learning engine using the static features.

2. The method according to claim 1, wherein the dividing the executable file into a plurality of to-be-extracted parts based on a data directory of the executable file comprises:

dividing the executable file into an executable file header part, an executable file section part, an executable file Overlay data part, and an executable file integral part based on the data directory of the executable file.

3. The method according to claim 1, wherein the extracting information entropy features of all the to-be-extracted parts and performing data transformation on the information entropy features to obtain static features comprises:

extracting the information entropy features of all the to-be-extracted parts, wherein the information entropy features are binary data;

calculating a proportion of each of characters in the binary data corresponding to the information entropy features to all the characters in the binary data, and obtaining an entropy flow of the information entropy features based on the proportion; and

performing a DCT transform or a DWT transform on the entropy flow to obtain the static features.

4. The method according to claim 3, wherein the calculating a proportion of each of characters in the binary data corresponding to the information entropy features to all the characters in the binary data and obtaining an entropy flow of the information entropy features based on the proportion comprises:

dividing the binary data corresponding to the information entropy features into a plurality of byte blocks, wherein each of the byte blocks comprises 256 bytes;

for each of the byte blocks, calculating a byte block proportion p of each of characters in the byte block to all characters in the byte block, and calculating a entropy flow component S for each of bytes in the byte block by using the following equation:

$$S = \left| p \cdot \log(p) \right|; \text{ and}$$

accumulating all the entropy flow components to obtain the entropy flow.

5. The method according to claim 1, wherein after obtaining the executable file, the method further comprises:

determining a file format of the executable file based on general format information of the executable file, and selecting a feature extraction strategy based on the file format;

the extracting information entropy features of all the to-be-extracted parts comprises:

extracting the information entropy features of all the to-be-extracted parts based on the feature extraction strategy.

6. The method according to claim 5, wherein the determining a file format of the executable file based on general format information of the executable file comprises:

determining values of .Net MetaData Directory RVA and .Net MetaData Directory Size in the data directory of the executable file; and

determining, if the values of the .Net MetaData Directory RVA and the .Net MetaData Directory Size are not empty, that the file format of the executable file is .Net format.

7. The method according to claim 1, wherein the dividing the executable file into a plurality of to-be-extracted parts based on a data directory of the executable file comprises:

dividing the executable file into a plurality of to-be-extracted parts based on a section array of the executable file.

8. The method according to any one of claims 1 to 7, wherein after training a machine learning engine using the static features, the method further comprises:

extracting, in a case that a suspicious software is detected, a target executable file of the suspicious software; and

inputting the target executable file into the trained machine learning engine to determine,

by the trained machine learning engine, whether the target executable file is an executable file of a malware.

9. The method according to claim 8, further comprising:

reporting, if it is determined by the trained machine learning engine that the target executable file is the executable file of the malware, software information of the suspicious software to perform manual detection on the suspicious software;

receiving a manual detection result, and determining whether the current determination by the trained machine learning engine is wrong based on the manual detection result; and

training, if the current determination by the trained machine learning engine is wrong, the machine learning engine using the target executable file as a new negative training sample.

10. A system for training a machine learning engine, comprising:

a file division module, configured to obtain an executable file and divide the executable file into a plurality of to-be-extracted parts based on a data directory of the executable file;

a static feature extraction module, configured to extract information entropy features of all the to-be-extracted parts and perform data transformation on the information entropy features to obtain static features; and

a training module, configured to train a machine learning engine using the static features.

11. The system according claim 10, wherein the file division module comprises:

a file obtaining unit, configured to obtain the executable file; and

a division unit, configured to divide the executable file into an executable file header part, an executable file section part, an executable file Overlay data part, and an executable file integral part based on the data directory of the executable file.

12. The system according claim 10, wherein the static feature extraction module comprises:

an information entropy feature extraction unit, configured to extract the information entropy features of all the to-be-extracted parts, wherein the information entropy features are binary data;

an entropy flow calculation unit, configured to calculate a proportion of each of characters in the binary data corresponding to the information entropy features to all the characters in the binary data and obtain an entropy flow of the information entropy features based on the proportion; and

a data transformation unit, configured to perform a DCT transform or a DWT transform on the entropy flow to obtain the static features.

13. The system according claim 12, wherein the entropy flow calculation unit comprises:

a division sub-unit, configured to divide the binary data corresponding to the information entropy features into a plurality of byte blocks, wherein each of the byte blocks comprises 256 bytes;

an entropy flow component calculation sub-unit, configured to, for each of the byte blocks, calculate a byte block proportion p of each of characters in the byte block to all characters in the byte block, and calculate a entropy flow component S for each of bytes in the byte block by using the following equation:

$$S = \left| p \cdot \log(p) \right|$$; and

an entropy flow component accumulation unit, configured to accumulate all the entropy flow components to obtain the entropy flow.

14. The system according claim 10, further comprising:

an extraction strategy determination module, configured to determine a file format of the executable file based on general format information of the executable file, and select a feature extraction strategy based on the file format;

the static feature extraction module is configured to extract the information entropy features of all the to-be-extracted parts based on the feature extraction strategy, and perform data transformation on the information entropy features to obtain the static features.

15. The system according claim 14, wherein the extraction strategy determination module comprises:

a file format determination unit, configured to determine values of .Net MetaData

Directory RVA and .Net MetaData Directory Size in the data directory of the executable file, and determine, if the values of the .Net MetaData Directory RVA and the .Net MetaData Directory Size are not empty, that the file format of the executable file is .Net format; and

a strategy selection unit, configured to select the feature extraction strategy based on the file format.

16. The system according claim 10, wherein the file division module is configured to obtain the executable file and divide the executable file into the plurality of to-be-extracted parts based on a section array of the executable file.

17. The system according any one of claims 10 to 16, further comprising:

an extraction module, configured to extract, in a case that a suspicious software is detected, a target executable file of the suspicious software; and

a determination module, configured to input the target executable file into the trained machine learning engine to determine, by the trained machine learning engine, whether the target executable file is an executable file of a malware.

18. The system according claim 17, further comprising:

a reporting module, configured to, if it is determined by the trained machine learning engine that the target executable file is the executable file of the malware, report software information of the suspicious software to perform manual detection on the suspicious software; and

a secondary training module, configured to receive a manual detection result, determine whether the current determination by the trained machine learning engine is wrong based on the manual detection result, and train, if the current determination by the trained machine learning engine is wrong, the machine learning engine using the target executable file as a new negative training sample.

19. A device for training a machine learning engine, comprising:

a memory, storing a computer program; and

a processor, configured to execute the computer program to perform the method for training a machine learning engine according to any one of claims 1 to 9.

20. A computer-readable storage medium storing a computer program, wherein the

computer program, when executed by a processor, causes the processor to perform the method for training a machine learning engine according to any one of claims 1 to 9.