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;

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 discrete cosine transform (DCT) or a discrete walsh transform (DWT) on the entropy flow to obtain 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 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.

4. 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.

5. The method according to claim 4, 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.

6. 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.

7. The method according to any one of claims 1 to 6, 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.

8. The method according to claim 7, 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.

9. 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;

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 discrete cosine transform (DCT) or a discrete walsh transform (DWT) on the entropy flow to obtain the static features.

10. The system according claim 9, 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.

11. The system according claim 9, 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.

12. The system according claim 9, 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.

13. The system according claim 12, 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.

14. The system according claim 9, 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.

15. The system according any one of claims 9 to 14, 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.

16. The system according claim 15, 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.

17. 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 8.

18. 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 8.