https://doi.org/10.31891/2307-5732-2025-353-1
УДК 004.773.2

БОЙКО МАКСИМ
Сумський державний університет
Національне антикорупційне бюро України
https://orcid.org/0000-0003-0950-8399
e-mail: mboiko25@gmail.com
МОСКАЛЕНКО В'ЯЧЕСЛАВ
Сумський державний університет
https://orcid.org/0000-0001-6275-9803
e-mail: v.moskalenko@cs.sumdu.edu.ua

УДОСКОНАЛЕНА МОДЕЛЬ КЛАСИФІКАЦІЇ БЛОКІВ БІНАРНИХ ДАНИХ ДЛЯ ЗАДАЧ КАРВІНГУ ФАЙЛІВ

У роботі розглянуто проблему класифікації блоків бінарних даних як складового етапу процесу карвінгу файлів із високим рівнем фрагментації. Існуючі моделі та методи мають високий рівень помилок у залежності від багатьох факторів. До того ж при вирішенні реальних задач дані, що аналізуються, можуть відрізнятися від представлених у навчальних датасетах. Метою цього дослідження є підвищити ефективність моделей класифікації блоків бінарних даних і подолати проблеми виявлення фрагментів нецільових типів файлів. У ході дослідження проведено удосконалення існуючих моделей ідентифікації фрагментів файлів. Удосконалені моделі передбачають введення додаткового відгалудження (голови) класифікатора, що відповідає за побудову прототипів класів у дискретному просторі ознак. Таким чином під час навчання досягається регуляризація простору ознак для класифікатора фрагментів цільових і нецільових типів файлів. При цьому будуються межі (контейнери) класів, що забезпечують виявлення даних, що виходять за межі навчального розподілу. У ході проведення експериментів із використанням пропонованих моделей вдалося отримати підвищення точності порівняно з базовими моделями від 1,9% до 3.1% в залежності від сценарії застосування. Загалом точність при ідентифікації фрагментів цільових типів файлів, розбитих на 5, 11 і 25 класів, становила від 88% до 98%, від 53% до 100% і від 72% до 100% відповідно. За результатами навчання було відмічено зростання відстані Хемінга між векторами-прототипами класів в бінарному просторі ознак у регуляризуючому відгалудженні класифікатора. Отримано такі макроусереднені значення валідаційної метрики якості класифікації F1 – 91,78%, 59,97% і 82,94% для сценаріїв із розбиттям простору на 5, 11 і 25 класів відповідно. Нижчі значення якості класифікації у сценарії із розбиттям файлів на 11 класів може бути наслідком наявності значного перетину класів у просторі ознак. Таким чином введення у запропонованих моделях регуляризуючого відгалудження моделі класифікатора дозволило отримати вищі точністні характеристик під час класифікації блоків бінарних даних, проте результати залежать від способу розбиття простору файлів на класи.

Ключові слова: штучний інтелект, машинне навчання, нейронна мережа, класифікація, ідентифікація, аналіз даних, набір даних, інформаційна технологія.

BOIKO MAKSYM
Sumy State University
The National Anti-corruption Bureau of Ukraine
MOSKALENKO VIACHESLAV
Sumy State University

IMPROVED BINARY DATA BLOCK CLASSIFICATION MODEL FOR FILE CARVING PROBLEMS

In this paper, we address the problem of classifying blocks of binary data as an integral stage in the file-carving process under conditions of high fragmentation. Existing models and methods exhibit high error rates depending on a variety of factors, and in real-world applications the data to be analyzed often differ from those found in training datasets. The aim of this study is to improve the effectiveness of binary-data-block classification models and to overcome the challenges involved in detecting fragments of non-target file types.

To that end, we have enhanced existing file-fragment identification models by introducing an additional classifier head responsible for constructing class prototypes in a discrete feature space. During training, this auxiliary branch serves to regularize the feature space for both target and non-target file fragments. At the same time, class-specific boundaries (or containers) are established to enable the detection of data that fall outside the training distribution.

Experimental evaluation of the proposed models demonstrated accuracy gains of 1.9 % to 3.1 % compared with baseline approaches, depending on the application scenario. Overall accuracy for identifying fragments of target file types partitioned into 5, 11, and 25 classes ranged from 88 % to 98 %, 53 % to 100 %, and 72 % to 100 %, respectively. Training also yielded an increase in the Hamming distance between prototype vectors in the binary feature space within the regularizing classifier head. The macro-averaged F1 scores were 91.78 %, 59.97 %, and 82.94 % for the 5-, 11-, and 25-class scenarios, respectively. The lower performance in the 11-class case appears to result from significant overlap between classes in the feature space. Thus, the introduction of a regularizing classifier branch in the proposed models leads to higher classification accuracy for binary-data blocks, although the results depend on the chosen class-partitioning scheme.

Keywords: artificial intelligence; machine learning; neural network; classification; identification; data analysis; dataset; information technology.

Стаття надійшла до редакції / Received 11.04.2025
Прийнята до друку / Accepted 26.04.2025

Introduction

A wide range of utilities and multifunctional software such as [1], [2]: Scalpel, Foremost, PhotoRec, Recoverit, X-Ways Forensics, FTK Forensic Toolkit, Autopsy, Magnet Axiom, UFS Explorer, are capable of recovering files by signatures. However, they are not effective for recovering highly fragmented data without file system metadata.

Recent tendencies in solving the problems of carving highly fragmented files suggest that this process should be divided into separate stages, such as [2]: identification of file fragments, clustering, reconstruction of files and/or their contents, and verification. Then, different information technologies are used for each of these stages, depending on the specifics of the recoverable data [3].

Classifying binary data blocks that do not have clearly defined markers is important for carving highly fragmented files with no file system metadata and in related areas [4]. Many researchers focus on the use of artificial intelligence techniques [2]. To reduce the impact of the human factor, machine learning models and methods with automatic feature selection are increasingly used to classify file fragments. The necessity to develop new models and improve existing ones in order to increase the efficiency of file fragment classification is an urgent task.

Review of the literature

An increasing number of works are focused on using models and methods of artificial intelligence for classifying binary data blocks [5], [6], [7], [8], [9], [10], [11]. Researchers choose models based on the support vector method, decision tree, and various types of neural networks as classifiers.

Some earlier works frequently used techniques with manual feature selection, such as the average value of unigrams and bigrams, their standard deviation, and the Hamming weight [5]. In this paper, the authors carried out experiments on classifying files divided into 14 classes and mapped to a hierarchical structure. The researchers used a classifier based on the support vector method. The classification was repeated several times at each level of the hierarchy. As a result, the average identification accuracy was 67.78%.

Another study [6] performed n-gram analysis using support vector machines and backpropagation neural networks. The authors achieved an identification accuracy of 65.12% to 91.49% for the support vector machines method and 85.8% to 85.88% for neural networks for the classification of file fragments divided into 6 classes.

In the paper [7], the authors propose a feature generation model using byte embeddings for feature extraction from 4096-byte fragments and k Nearest Neighbors for identification. In this way, the researchers improved the accuracy and precision by 3% to 12% compared to other feature extraction techniques and classifiers discussed in the paper. The average accuracy rate was 72%.

Instead, the use of neural networks in combination with automatic feature extraction was demonstrated to be a perspective direction [8], [9], [10], [11].

For example, in [8], the authors transformed data blocks of 4096 bytes into a 64x64 grayscale image. The authors obtained an accuracy rate of 70.9% when classifying 16 file types.

In [9], a series of experiments were conducted with the training dataset FFT-75 [12]. The authors tested six different scenarios of dividing file space into classes for data blocks of 512 bytes and 4096 bytes. The proposed models were based on the use of several one-dimensional convolutional layers. Inputs were all byte values from the data block. The average accuracy of fragment identification was 65.6 %, 78.9 %, 87.9 %, 90.2 %, and not less than 99.0 %, depending on the number of classes: 75, 11, 25, 5, and 2, respectively.

Similar experiments on the same dataset were conducted in [10], where a deepwise separable convolutional neural network was used. Compared to [9], the authors achieved better results only when identifying file fragments divided into 75 classes (66.33% and 79.27%). In all other cases, the results were 0.2-7.2% worse.

Paper [11] proposes an approach involving three additional self-attention modules. As a result, significant bytes in the data fragment and contextual information from contiguous sectors are taken into account. The accuracy of file fragment identification is better in 4 out of 6 scenarios when compared to the Govdocs1 dataset [13] and in 5 out of 6 scenarios for the DFRWS2006 dataset [14].

Therefore, the efficiency of artificial intelligence models and algorithms for classifying binary data by file type depends on many aspects, such as the type and architecture of the classifier, the type and method of feature selection, training datasets, and the type and method of file space dividing into classes. Most models and algorithms need to improve their effectiveness. In addition, there are high error rates when recognizing out-of-distribution data and testing approaches on other datasets.

The purpose and tasks of research

The object of the study is the process of classifying file fragments as an essential stage of file carving.

The subject of the study is models and methods of information technology for identifying binary data blocks.

The goal of the study is to improve the efficiency of models for classifying binary data blocks and overcome the problem of identifying fragments of non-target file types.

To achieve this goal, we need to solve the following tasks:

- select a model for identifying binary data blocks;
- propose changes to the architecture of the selected model to improve its efficiency and effectiveness

in identifying fragments of target and non-target file types;

- conduct experiments and analyze the results.

Classification of file fragments as part of the process of carving highly fragmented files

The most difficult cases are recovering highly fragmented files with three or more fragments [2], [3],[15]. The reason for this is that middle data blocks usually don't have well-defined markers, such as the header and footer signatures. This task can become even more difficult if the fragments of the target files are out of order. In this case, detecting and identifying the middle fragments in the file to which they belong is difficult. The first stage of file carving - identifying binary data blocks - requires finding all available fragments of such files. After that, the detected file fragments should be organized in the correct order, i.e., the file and/or part of its contents should be reconstructed.

Thus, identifying binary data blocks is actually part of an intelligent system for carving highly fragmented files. In general, such a system can be visualized in the form of diagrams shown in Figs. 1, 2. The above figures show a generalized and detailed functional model of the carving process for highly fragmented files, where input and output data, control information, and mechanisms involved are indicated. Implementing this system requires using a set of tools and methods to create models and methods for data analysis and processing.

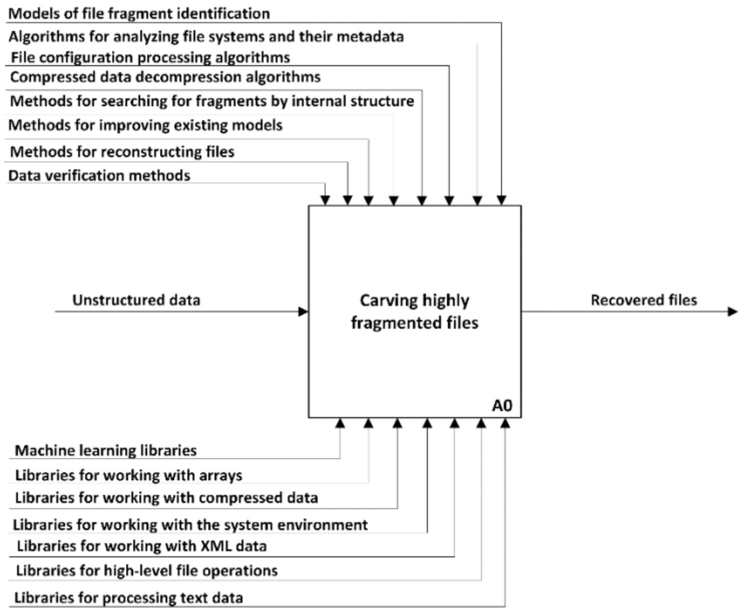

Models of file fragment identification
Algorithms for analyzing file systems and their metadata
File configuration processing algorithms
Compressed data decompression algorithms
Methods for searching for fragments by internal structure
Methods for improving existing models
Methods for reconstructing files
Data verification methods
Unstructured data
Carving highly fragmented files
A0
Recovered files
Machine learning libraries
Libraries for working with arrays
Libraries for working with compressed data
Libraries for working with the system environment
Libraries for working with XML data
Libraries for high-level file operations
Libraries for processing text data

Fig. 1 Generalized functional model of the carving process for highly fragmented files

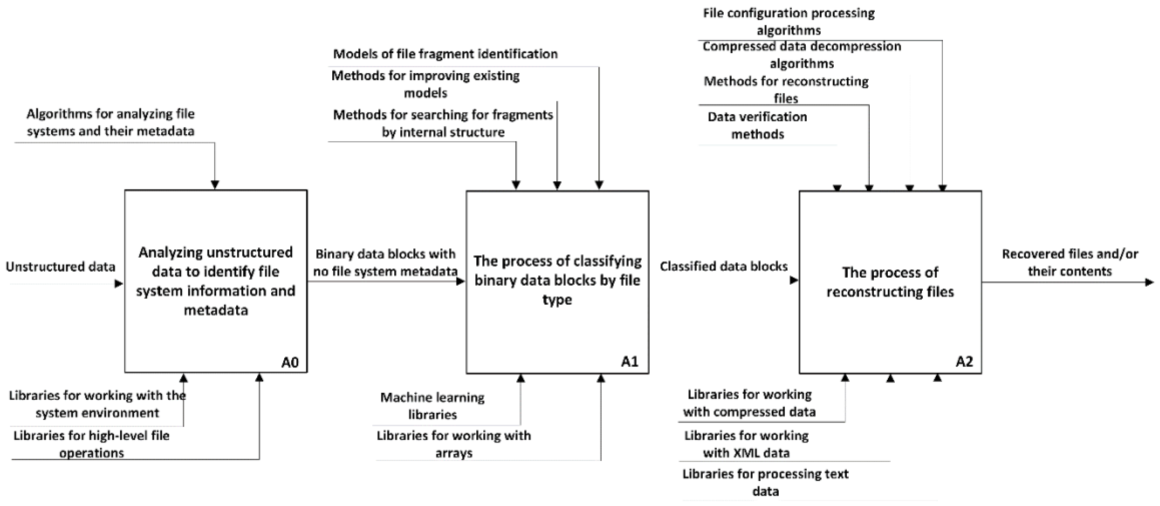

Algorithms for analyzing file systems and their metadata
Unstructured data
Analyzing unstructured data to identify file system information and metadata
A0
Libraries for working with the system environment
Libraries for high-level file operations
Binary data blocks with no file system metadata
Models of file fragment identification
Methods for improving existing models
Methods for searching for fragments by internal structure
The process of classifying binary data blocks by file type
A1
Machine learning libraries
Libraries for working with arrays
Classified data blocks
File configuration processing algorithms
Compressed data decompression algorithms
Methods for reconstructing files
Data verification methods
The process of reconstructing files
A2
Recovered files and/or their contents
Libraries for working with compressed data
Libraries for working with XML data
Libraries for processing text data

Fig. 2 Detailed functional model of the carving process of highly fragmented files

Criteria for evaluating the classification of binary data blocks

In general, the problem of identifying file fragments is to find a function f that classifies n-byte data blocks B by file type labels T [9]:

$$f : B \to T \tag{1}$$

where $B \in \mathbb{Z}^{n}_{255}$;

T ; $\in \{PDF, DOCX, \dots, PNG\}$

$\mathbb{Z}_{255} = [0, \ldots, 255]$;

n is the size of the data block.

The alphabet of classes T has a power K, where the value of K depends on the number of target file types. $\mathbb{Z}_{255}$ is the alphabet of all possible byte values in the data block B. This alphabet consists of values from 0 to 255 and has a power of 256. The value of the data block size n is usually equal to the standard sector and cluster sizes for many file systems - 512 or 4096 bytes.

To evaluate the effectiveness of the classification process, we used the criteria of accuracy, precision, recall, and F-measure. To do this, the number of correctly classified (true positive or TP), incorrectly classified (false positive or FP), correctly rejected (false positive or FP), and unclassified (false negative or FN) file fragments were counted [10], [16].

The accuracy was defined as follows [10]:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

The precision was calculated using the following formula [16]:

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

The recall was calculated as follows [16]:

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

The F-measure was used to determine the overall performance [16]:

$$F - measure = \frac{1}{\frac{\alpha}{P} + \frac{1 - \alpha}{R}} \tag{5}$$

where P is precision,

R – recall,

α is a numerical value from 0 to 1 to determine precision and recall weights.

The architecture of a neural network model for classifying file fragments

In order to overcome the problem of detecting fragments of non-target files when identifying binary data blocks, the paper proposes to improve existing models described in [9]. For this purpose, it is proposed to introduce a regularization classification head into the neural network [17], [18]. The improved model has two classifier heads and uses the concept of class prototypes in a discrete feature space [17], [18]. Its first head is responsible for the standard classification among target file fragments and determines the fragments belonging to target file types to a particular class. Instead, the second chapter is designed to ensure the building of class containers and effectively determine whether a fragment belongs to a particular class.

As a basis, binary data block identification models with an automatic feature extractor [9] were used to classify 512-byte file fragments categorized into classes 5, 11, and 25. The inputs to the models were data blocks as a sequence of bytes. The baseline models have the following sequence of layers: Embedding, 1D-CNN, MaxPooling1D, 1D-CNN, MaxPooling1D, GlobalAveragePooling1D, Dropout, Dense, Dense. The Prototype layer is added to the baseline models as the second output head (Fig. 3). The values of some model hyperparameters, such as output_dim, kernel_size, pool_size, and nodes, differed depending on the chosen classification scenario. In Fig. 3, these values are shown in curly brackets for the cases of identifying file fragments divided into 5, 11, and 25 classes, respectively.

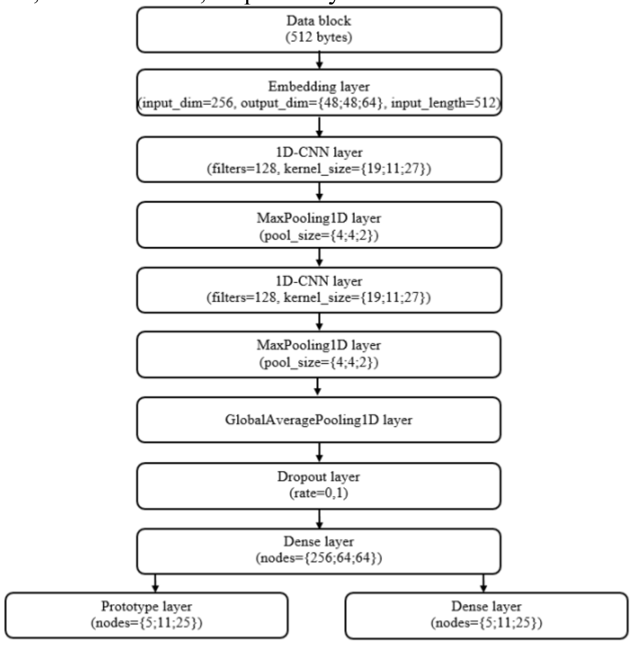


Fig. 3 Architecture of the proposed two-head classification models

The proposed model determines the membership of a binary data block B to the container of the k-th class as follows [18]:

$$\mu_k(z) = 1 - \frac{dist(z,\ \bar{z}_k)}{N \cdot r_k} \tag{6}$$

where z is the binarized feature representation of B,
$\bar{z}_k$ – K-class prototype,
N is the dimension of the feature representation,
$r_k \in (0; 1)$ is the scale coefficient for a k-class container,
$dist(\cdot)$ is the Squared Euclidean distance.

The learning process is similar to standard classification methods. However, this process differs in that the cross-entropy function is used. This allows to minimize the difference between the probability distributions of model predictions and true class labels. The difference between the one-hot coded class label y and the normalized value of the membership function μ is calculated using the following formula [18]:

$$loss\_prototype = CE(Softmax(\mu), y) \tag{7}$$

where μ is the normalized value of the membership function,
Softmax(μ) is the function for converting μ values into probabilities for each class.
CE is the cross-entropy function.

A discretization error penalty is applied to reduce the negative impact of noisy and less important data and focus the classification model on the most relevant features [18]. This penalty can be calculated as follows:

$$L_D = z^{\mathrm{T}}(e - z) \tag{8}$$

where z is the vector of model outputs,
e is a unit vector with a dimension equal to the number of classes, each element of which equals 1.

Experiments

The study included a number of experiments using the FFT-75 dataset [12]. The experiments involved training of models with the architectures shown in Fig. 3. Three scenarios were considered [9], which differed in the way the file space was divided into classes.

In Scenario 1, the array of file fragments is divided into 5 classes: JPEG files; RAW images (3FR, NEF, etc.); video files (AVI, MKV, etc.); other graphic files (TIFF, HEIC, BMP, GIF, and PNG); other file types.

Scenario 2 included 11 classes of files: graphic files (JPEG, TIFF, HEIC, BMP, GIF, and PNG); RAW images (3FR, NEF, etc.); vector files (AI, EPS, and PSD); video files (AVI, MKV, etc.); archive files (RAR, ZIP, etc.); executable files (EXE, MACH-O, ELF, and DLL); office files (DOC, DOCX, KEY, PPT, PPTX, XLS, and XLSX); published files (DJVU, PDF, MOBI, and EPUB); human-readable files (RTF, TXT, CSV, LOG, etc.); audio files (M4A, MP3, etc.); other file types (PCAP, TTF, DWG, and SQLITE).

In scenario 3, the files were divided into 25 classes: six classes of graphic files (JPG, TIFF, HEIC, BMP, GIF, and PNG); eleven classes of RAW images (ARW, CR2, DNG, GPR, NEF, NRW, ORF, PEF, RAF, RW2, and 3FR); seven classes of video files (MOV, MP4, 3GP, AVI, MKV, OGV, and WEBM); one class that contains all other files.

Each model was trained for 120 epochs. The accuracy reached a plateau in all three scenarios after approximately 100-105 training epochs. The graphs illustrating the variations in the accuracy of binary data block identification and classification of target and non-target file fragments depending on the number of epochs on the test set for each head are shown in Fig. 4 and 5. The average accuracy of the binary data block classification for the proposed models were 92.1%, 79.9%, and 91.0% for scenarios 1, 2, and 3, respectively. At the same time, the macro-average values of F1 measures were 91.9%, 80%, and 91.0%, respectively.

Figs. 6 and 7 show the confusion matrices that demonstrate the results of applying the proposed models for the standard classification of binary data blocks in scenarios 1, 2, and 3, respectively. The horizontal scale is accountable for the predicted labels, and the vertical scale is accountable for the true labels.

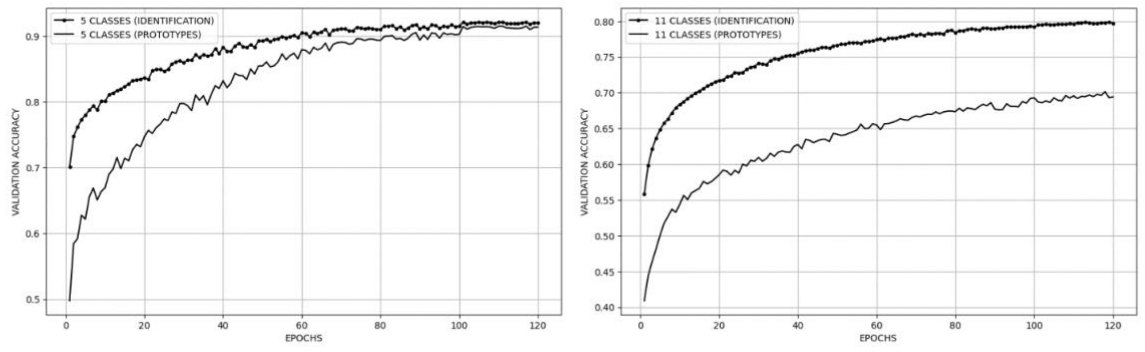


Fig. 4 Graph of the identification accuracy and the classification of target and non-target file fragments versus the number of training epochs ((a) Scenario 1 and (b) Scenario 2)

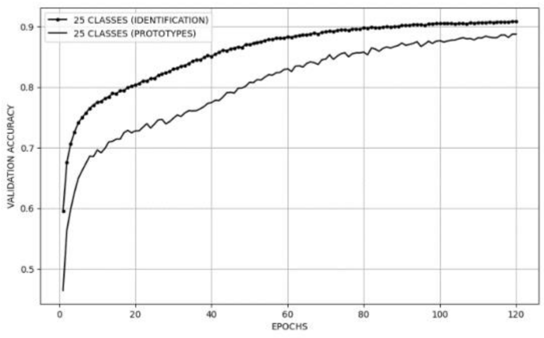
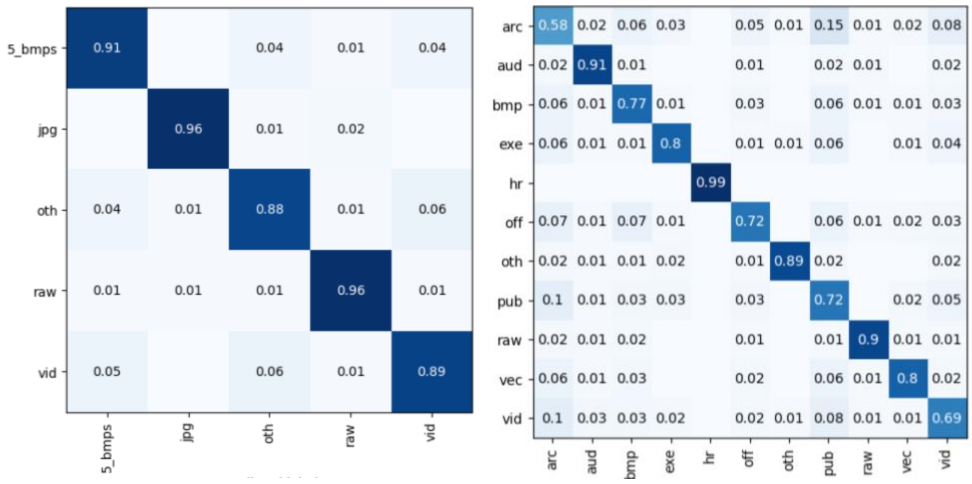


Fig. 5 Graph of the identification accuracy and the classification of target and non-target file fragments versus the number of training epochs (Scenario 3)



a b

Fig. 6 Confusion matrices: (a) Scenario 1; (b) Scenario 2

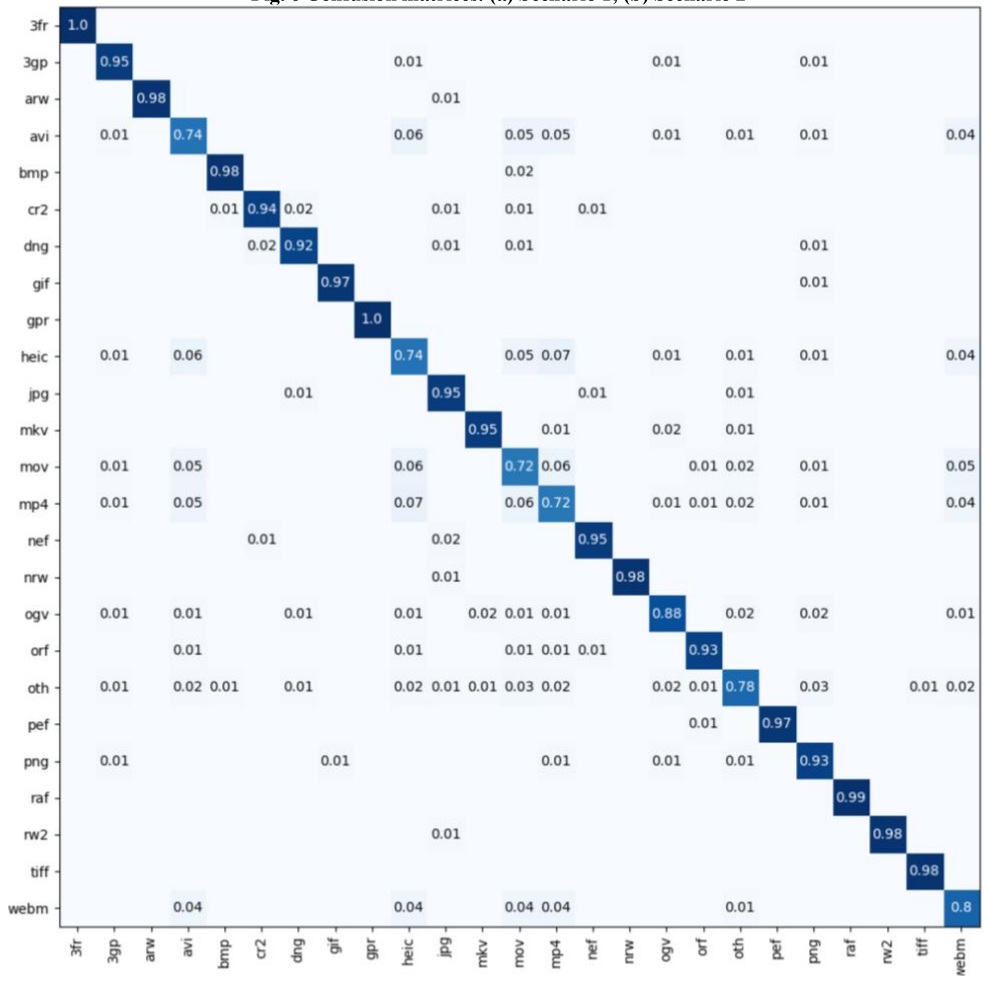


Fig. 7 Confusion matrix (Scenario 3)

To evaluate the results of classifying target and non-target file fragments, a number of metrics were calculated based on the training data, as shown in Table 1.

Table 1

Evaluation of the model in terms of classification of fragments of target and non-target files

Parameter	Scenario 1	Scenario 2	Scenario 3
Maximum radiuses for each class (on regularization classification head)	[101, 103, 92, 87, 87]	[17, 15, 22, 15, 12, 21, 19, 12, 39, 12, 22]	[27, 23, 24, 20, 22, 24, 27, 26, 25, 27, 24, 27, 20, 17, 20, 22, 20, 21, 21, 20, 17, 21, 21, 18, 24]
Optimized radii for each class (on regularization classification head)	[67, 66, 49, 52, 38]	[9, 10, 13, 8, 11, 12, 11, 11, 28, 4, 14]	[20, 16, 18, 18, 15, 16, 15, 19, 17, 19, 16, 18, 13, 15, 16, 14, 14, 18, 17, 13, 14, 15, 14, 12, 20]
Efficiency criterion	[0,96; 0,98; 0,97; 0,96; 0,96]	[0,25; 0,55; 0,59; 0,40; 0,03; 0,59; 0,49; 0,33; 0,97; 0,55; 0,67]	[0,46; 0,88; 0,54; 0,04; 0,87; 0,60; 0,84; 0,53; 0,75; 0,91; 0,87; 0.92; 0.73; 0.06; 0.82; 0.69; 0.57; 0.21; 0.14; 0.60; 0.27; 0.62; 0.41; 0.48; 0.11]
Minimum distance between classes (on regularization classification head)	87	12	17
Maximum distance between classes (on regularization classification head)	174	54	44
Average distance between classes (on regularization classification head)	120,6	26,5	30,6
Macro-averaged value of the F1 measure	91,78%	59,97%	82,94%
Micro-average value of the measure F1	91,78%	64,21%	82,77%

Discussion

The results demonstrate the suitability of the proposed models for classifying file fragments. Table 2 shows the average accuracy of the binary data block classification for the baseline [9], models described in papers [10] and [11], and the proposed models.

Table 2

Average accuracy comparison on test sets

Number of classes	Accuracy (baseline models)	Accuracy [10]	Accuracy [11]	Accuracy (proposed models)
5	90,2%	87,3%	93,6%	92,1%
11	78,9%	75,8%	90,4%	79,9%
25	87,9%	80,8%	93,5%	91,0%

The analysis of the results in Table 2 confirms the greater efficiency of the proposed models in identifying binary data blocks in all three scenarios compared to the baseline models and models examined in the research [10]. The average accuracy values were 92.1%, 79.9%, and 91.0% and were 1.9%, 1%, and 3.1% higher compared to the cases when the baseline models were used [9]. This is achieved by regularizing the feature space with an additional classification head. This makes the model more robust. Instead, the proposed models showed slightly worse results for scenarios 1 and 3 compared to the models of [11] – by 1,5% and 2,5%, respectively. Although the proposed models did not outperform the results of [11], it can be assumed that the use of the proposed approach to the models of [11] may also have a positive impact on their effectiveness.

The classification results presented in Figures 6 and 7 show some aspects that can be focused on to improve the efficiency of the proposed models. As shown in Figure 6a and Table 1, in Scenario 1, the accuracy rates for various classes range from 89% to 96%. In general, Figs. 6 and 7 illustrate that these models are all quite capable of classifying RAW images, graphics, and video files. Worse results (72% to 80%) were obtained only in Scenario 3 for such file types as HEIC, AVI, MOV, MP4, and WEBM. However, as can be seen from Fig. 7, 22% of HEIC file fragments, 20% of AVI file fragments, 22% of MOV file fragments, 22% of MP4 file fragments, and 16% of WEBM file fragments are incorrectly identified as parts of other files within the same group. That is, the classes that include fragments of HEIC, AVI, MOV, MP4, and WEBM files are similar.

As for Scenario 2, the classes of the archive, video, office, graphic, and published document files have accuracy values of 58% to 77% (Fig. 6b). Just like in the case of HEIC, AVI, MOV, MP4, WEBM files, the reason for this can be explained by the similar structure of file types in these classes and the presence of third-party file types in compound files such as DOCX, DOC, PDF, PPT, PPTX, etc.

Prototype classes were built for each scenario to evaluate the proposed model's effectiveness in identifying fragments of target and non-target file types. Scenario 1's prototype length was 256 elements, for scenarios 2 and 3 – 64. As can be seen from Tables 2, 3, and 4, the smallest, largest, and average distances between any pair of prototypes were 87, 174, and 120.6 (Scenario 1), 12, 54, and 26.5 (Scenario 2), and 17, 44, and 30.6 (Scenario 3). If the prototype classes were reduced to a single dimension of 64 elements, then Scenario 1 would have the following indicators of the smallest, largest, and average distances: 21.75, 43.5, and 30.15, respectively. These results indicate the presence of similar classes in scenario 2. This also correlates with the calculated F1 measure and the results of the standard classification.

As can be seen from Table 1, the efficiency measures for Scenario 1 have values between 96% and 98%. This also correlates with the values from Fig. 6a and indicates a good classification ability. As can be seen from Table 1, in Scenario 2, the performance values for graphic, archive, and published files are 25%, 3%, and 33%. Thus, this model has problems with these file types. Similar conclusions can be drawn about HEIC, MOV, MP4, AVI, DNG files, and the class that includes all other file types in Scenario 3 (Table 1). These results also correlate with the data in Figs. 6 and 7. Similarly to standard classification, the models demonstrate better results in Scenarios 1 and 3. In fact, the efficiency metrics vary depending on the way and quality of the file space dividing into classes.

Thus, the proposed models show better overall performance compared to the baseline ones. In addition, such models can detect file fragments that do not belong to any defined classes.

Conclusions from this study and prospects for further research in this direction

This paper presents an improved model for classifying binary data blocks for file carving tasks in order to increase its efficiency and ability to work on out-of-distribution data. The proposed improvement was implemented by introducing a of a regularization classification head into the neural network responsible for building class containers and employing the principle of error-correcting coding. In fact, introducing an additional head at the model output allows for the regularization of the feature space to increase robustness.

It was experimentally confirmed that the proposed models with a two-head classifier are able to identify binary data blocks with better performance compared to the baseline models. Thus, the average classification accuracy rates were 79.9%, 91.0%, and 92.1% in different scenarios. These values were higher by 1.9%, 1.0%, and 3.1%, respectively, compared to the baseline models. Overall, the classification accuracy of various file types was 88% to 98%, 53% to 100%, and 72% to 100% for scenarios 1, 2, and 3, respectively.

At the same time, the proposed models can detect fragments of non-target file types, taking into account the container boundaries. The experiments obtained the following values of macro- and micro-averaged indicators of the F1 measure for scenarios 1, 2, and 3, respectively: 91.78% and 91.78%; 59.97% and 64.21%; 82.94% and 82.77%. In general, the created file prototypes vary from one another. However, in scenarios 2 and 3, there are more similar classes.

Future research will focus on creating a fully functional system for carving highly fragmented files.

Contribution of authors: development of the model, conducting experiments, analysis of the results, visualization of the results – **M. V. Boiko**; supervision and editing of work – **V. V. Moskalenko**.

Acknowledgments: the authors thank the Ministry of Education and Science of Ukraine for the support to the Laboratory of Intellectual Systems in the framework of research project No. 0124U000548 "Information Technology for Ensuring the Resilience of the Intelligent Onboard System of Small-Sized Aircraft" (2024–2026).

References

1. Yuwono, D. T., Fadlil, A., & Sunardi, S. (2019). Performance Comparison of Forensic Software for Carving Files using NIST Method. *Jurnal Teknologi Dan Sistem Komputer*, *7*(3). https://doi.org/10.14710/jtsiskom.7.3.2019.89-92.

2. Boiko, M., Moskalenko, V., & Shovkoplias, O. (2023). Advanced file carving: ontology, models and methods. *Radioelectronic and Computer Systems*, *3 (107)*, 204–216. https://doi.org/10.32620/reks.2023.3.16.

3. Ramli, N. I. S., Hisham, S. I., & Razak, M. F. A. (2021). Survey of File Carving Techniques. In *Lecture Notes on Data Engineering and Communications Technologies* (Vol. 72). https://doi.org/10.1007/978-3-030-70713-2_74.

4. LYSENKO, S., ATAMANIUK, O., BOKHONKO, O., & VOROBIYOV, V. (2023). METHOD FOR DETECTION OF RANSOMWARE CYBER THREATS BASED ON HONEYPOT: STATE-OF-ART. *Herald of Khmelnytskyi National University. Technical Sciences*, *317*(1), 300–309. https://doi.org/10.31891/2307-5732-2023-317-1-300-309.

5. Bhatt, M., Mishra, A., Kabir, M. W. U., Blake-Gatto, S. E., Rajendra, R., Hoque, M. T., & Ahmed, I. (2020). Hierarchy-Based File Fragment Classification. *Machine Learning and Knowledge Extraction*, *2*(3). https://doi.org/10.3390/make2030012.

6. Sester, J., Hayes, D., Scanlon, M., & Le-Khac, N. A. (2021). A comparative study of support vector machine and neural networks for file type identification using n-gram analysis. *Forensic Science International: Digital Investigation*, *36*. https://doi.org/10.1016/j.fsidi.2021.301121.

7. Haque, M. E., & Tozal, M. E. (2022). Byte embeddings for file fragment classification. *Future Generation Computer Systems*, *127*, 448–461. https://doi.org/10.1016/j.future.2021.09.019.

8. Chen, Q., Liao, Q., Jiang, Z. L., Fang, J., Yiu, S., Xi, G., Li, R., Yi, Z., Wang, X., Hui, L. C. K., Liu, D., & Zhang, E. (2018). File fragment classification using grayscale image conversion and deep learning in digital forensics. *Proceedings - 2018 IEEE Symposium on Security and Privacy Workshops, SPW 2018*. https://doi.org/10.1109/SPW.2018.00029.

9. Mittal, G., Korus, P., & Memon, N. (2021). FiFTy: Large-Scale File Fragment Type Identification Using Convolutional Neural Networks. *IEEE Transactions on Information Forensics and Security*, *16*, 28–41.

10. Saaim, K. M., Felemban, M., Alsaleh, S., & Almulhem, A. (2022). Light-Weight File Fragments Classification Using Depthwise Separable Convolutions. *IFIP Advances in Information and Communication Technology*, *648 IFIP*. https://doi.org/10.1007/978-3-031-06975-8_12.

11. Liu, W., Wang, Y., Wu, K., Yap, K. H., & Chau, L. P. (2023). A Byte Sequence is Worth an Image: CNN for File Fragment Classification Using Bit Shift and n-Gram Embeddings. *AICAS 2023 - IEEE International Conference on Artificial Intelligence Circuits and Systems, Proceeding*. https://doi.org/10.1109/AICAS57966.2023.10168636.

12. Mittal, G., Korus, P., & Memon, N. (2019). File Fragment Type (FFT) - 75 Dataset. *IEEE Dataport,* https://doi.org/10.1109/TIFS.2020.3004266.

13. Garfinkel, S., Farrell, P., Roussev, V., & Dinolt, G. (2009). Bringing science to digital forensics with standardized forensic corpora. *Digital Investigation*, *6*(SUPPL.). https://doi.org/10.1016/j.diin.2009.06.016.

14. *Forensic Challenges - DFRWS*. (n.d.). Https://Dfrws.Org/Forensic-Challenges/.

15. Suthar, H., & Sharma, P. (2024). An Investigation on File Carving Tool Methodologies Using Scenario Based Image Creation. *Indian Journal of Science and Technology*, *17*(3), 215–227.

16. Alghafli, K., Jones, A., & Martin, T. (2014). Investigating and measuring capabilities of the forensics file carving techniques. *Lecture Notes in Electrical Engineering*, *276 LNEE*. https://doi.org/10.1007/978-3-642-40861-8_47.

17. Moskalenko, V., & Moskalenko, A. (2022). NEURAL NETWORK BASED IMAGE CLASSIFIER RESILIENT TO DESTRUCTIVE PERTURBATION INFLUENCES – ARCHITECTURE AND TRAINING METHOD. *Radioelectronic and Computer Systems*, *2022*(3). https://doi.org/10.32620/reks.2022.3.07.

18. Boiko, M., & Kudryavtsev, A. (2023). Neural classifier of file fragments – model and training method. *International Scientific and Technical Conf. Informatics. Mathematics. Automation, IMA-2023*, 133–134.