International Journal of Safety and Security Engineering
Vol. 16, No. 7, July, 2026, pp. 1549-1561
Journal homepage: http://iieta.org/journals/ijsse

Enhanced File Fragment Classification Using Convolutional Neural Networks with Channel and Spatial Attention Mechanisms for Digital Forensics

Ayoub Tazi[1*], Ahmed El-Yahyaoui[1], Iyad Lahsen-Cherif[2]

[1] Intelligent Processing and Security of Systems (IPSS), Faculty of Sciences, Mohammed V University in Rabat, Rabat 10000, Morocco
[2] National Institute of Posts and Telecommunication (INPT), Rabat 10100, Morocco

Corresponding Author Email: Ayoub_tazi@um5.ac.ma

https://doi.org/10.18280/ijsse.160712

Received: 5 June 2026
Revised: 18 July 2026
Accepted: 24 July 2026
Available online: 31 July 2026

Keywords:
attention mechanisms, binary analysis, Convolutional Block Attention Module, Convolutional Neural Networks, digital forensics, file fragment classification

ABSTRACT

The task of file fragment classification is highly important in digital forensics, especially when file headers are corrupted or absent. In such cases, traditional classification methods based on statistical measures combined with machine learning algorithms remain useful. Despite the wide range of previously developed approaches relying on statistical analysis and machine learning, deep learning models have recently demonstrated improved performance. While recent approaches have explored Vision Transformer-based models and ultralight architectures, in this work, we introduce a Convolutional Neural Network (CNN) model integrating the Convolutional Block Attention Module (CBAM) in order to balance classification performance, interpretability, and computational efficiency. Specifically, we propose combining channel and spatial attention to distinguish between file types with similar characteristics, such as high-entropy and composite formats. We evaluate our approach on sixteen file types from the GovDocs dataset using a source-file-level split to prevent fragments from the same original file from appearing in different subsets. The revised model achieves an accuracy of 78.43%, a balanced accuracy of 76.00%, a macro-F1 score of 72.48%, and a weighted-F1 score of 77.85% on 227,205 held-out test fragments. Compared with Transformer-based dual-branch architectures, which may be more difficult to interpret, the proposed CNN-CBAM architecture provides class-specific attention-map visualizations while maintaining competitive performance with reduced complexity.

1. INTRODUCTION

Digital forensics often faces scenarios where file systems may be corrupted, damaged, or intentionally tampered with. As a result, conventional file identification approaches are rendered ineffective; therefore, forensic investigators have to resort to file fragment classification techniques to analyze raw binary data without relying on filenames, extensions, or metadata. The increasing complexity of file formats and the data obfuscation practices used by malicious actors has compounded the problem. File fragment identification entails analyzing binary fragments to infer the originating file types even in the absence of reliable header information.

The proposed approach attempts to address some of the limitations of existing methods by introducing a Convolutional Neural Network (CNN)-Convolutional Block Attention Module (CBAM) model, thereby enabling improved classification and interpretation of file fragments. In contrast to merely adding an attention layer to the network, the proposed solution demonstrates how attention can be decomposed into components that are meaningful from a forensic perspective: channel attention supports the identification of discriminative byte-value patterns, while spatial attention helps detect discriminative regions within the fragment representation ("what" and "where" information, respectively). This work builds on the previous concept of converting binary files into grayscale images, while using a stricter source-file-level evaluation protocol to reduce the risk of fragments from the same original file appearing in different subsets.

The main contributions of this research can be stated as follows:

(1) Design of a CNN-CBAM architecture specifically adapted to file fragment classification.

(2) A reproducible experimental evaluation of the CNN-CBAM model on sixteen file types from the GovDocs dataset, using a source-file-level split, a fixed random seed, and explicitly reported training, validation, and test fragment counts.

(3) A balanced mini-batch training strategy designed to reduce the effect of class imbalance without modifying the proposed CNN-CBAM architecture.

(4) A comprehensive evaluation of CNN-CBAM performance using accuracy, balanced accuracy, macro-F1, weighted-F1, per-class metrics, normalized confusion matrices, and CBAM attention maps.

Table 1 illustrates classical file signatures; however, the proposed protocol skips the first 512 bytes of each source file before fragment extraction. The complete source-file-level preprocessing and CNN-CBAM classification workflow is summarized in Figure 1.

Both the classification performance and interpretability of the proposed model are important for forensic analysis because they contribute to more reliable fragment-type identification, including in challenging cases involving high-entropy or compressed content. In addition, the model supports the interpretation of its decision-making process through visual attention maps that indicate which fragment regions influence the classification result.

The rest of the paper is organized as follows. Section 2 presents a review of relevant literature regarding file fragment classification and attention mechanisms. Section 3 presents some basic concepts such as CNN, binary-to-image conversion, and CBAM. Section 4 presents the methodology used to perform the classification, which includes dataset preparation, model design, and training configuration. Section 5 discusses the results obtained. Section 6 presents the discussion part of the paper. Finally, Section 7 concludes the paper and suggests future research directions.

Table 1. Common file signatures used in classical identification

Name	Binary Data (Hex)	Size	Description
JPEG Header	FF D8 FF E0	4	Magic number indicating the start of a JPEG file.
PDF Header	25 50 44 46 2D 31 2E	7	PDF file signature including version information.
PNG Signature	89 50 4E 47 0D 0A 1A 0A	8	Fixed PNG signature composed of magic bytes and control values.
ZIP Local Header	50 4B 03 04	4	ZIP archive start marker.
NOP Sled	90 90 90 90	4	No-operation instructions used in x86 binaries.

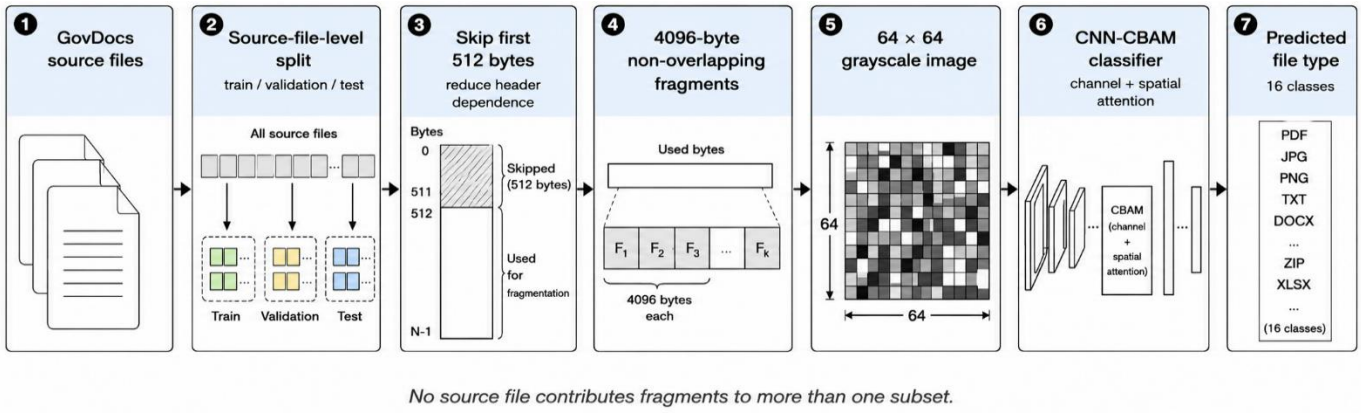

Figure 1. Overview of the proposed file fragment classification pipeline

2. RELATED WORK

Several methods have been proposed for file type and file fragment identification in digital forensics. Previous studies mostly focused on intact files or file fragments for which sufficient statistical information was still available, whereas more recent work has explored deep learning techniques that can operate directly on raw byte representations. Overall, the existing literature can be organized into four main research directions: traditional statistical methods, deep learning approaches, attention-based mechanisms, and class imbalance handling strategies.

This organization allows the comparison to focus not only on reported accuracy, but also on the dataset, fragment size, number of classes, computational complexity, and interpretability.

2.1 Traditional statistical approaches

Traditional file fragment classification techniques are primarily based on statistical features extracted from raw byte streams. Byte frequency distribution (BFD), byte frequency cross-correlation (BFC), entropy, n-grams, and compression-based similarities are commonly used to represent the binary structure of file fragments. These techniques are attractive because of their simplicity, interpretability, and lack of dependence on large training datasets. However, their effectiveness deteriorates when fragments are short, compressed, encrypted, or when header information is absent or deliberately altered.

McDaniel and Heydari [1] proposed content-based approaches for classifying file types using BFD and BFC, reporting accuracies of up to 27% and 49%, respectively. Their work also showed that accuracy may be much higher when magic numbers are available, which highlights the importance of header information. Nevertheless, in practical forensic cases, headers may be absent, corrupted, or deliberately altered. Beebe et al. [2] later used concatenated n-gram vectors with linear SVM classifiers and achieved 73.45% accuracy in experiments involving both public and proprietary datasets. Other approaches, such as PCA combined with neural networks [3], used reduced byte-frequency representations to build fileprints for file type detection.

NLP-inspired approaches have also been applied to represent fragments as byte sequences. In particular, unigram byte counts, bigram byte counts, and entropy were used as features in a bag-of-bytes framework for fragment classification [4]. Although such feature-based techniques remain useful, they rely on handcrafted descriptors and may struggle with high-entropy data, composite formats, and

fragments containing limited discriminative information.

2.2 Deep learning approaches

Deep learning methods aim to reduce the dependence on handcrafted features by learning discriminative patterns directly from byte-level data. A common strategy consists of converting binary fragments into grayscale images, where each byte is represented as one pixel. This representation allows CNNs to exploit spatial patterns in byte sequences while keeping the input format simple.

Chen et al. [5] transformed file fragments into grayscale images and used CNNs for classification, reporting an accuracy of 70.9% on a GovDocs-based 16-class setting with 4096-byte fragments. This work helped establish the relevance of CNNs for file fragment classification and showed that binary-to-image conversion can capture useful byte-level structure. CNN-based fragment classification was also explored [6], further supporting the ability of convolutional models to learn patterns in byte streams without extensive manual feature engineering.

Recent work has also investigated architectures designed to capture sequential or fine-grained byte relationships. CNN-LSTM models combine convolutional feature extraction with recurrent layers in order to model sequential dependencies among byte patterns [7]. However, recurrent processing can limit parallel execution during inference. ByteNet introduced a dual-branch Vision Transformer approach with a 1-bit sliding window mechanism to capture intra-byte relationships [8]. Although such models can represent richer bit-level patterns, their dual-branch and Transformer-based design may increase architectural complexity. In contrast, lightweight CNN approaches such as M-DSC use depthwise separable convolutions and squeeze-and-excitation blocks to reduce model size and improve inference efficiency, especially for resource-constrained forensic environments [9].

These studies show that recent deep learning models differ not only in terms of reported accuracy, but also in dataset characteristics, fragment size, number of classes, computational cost, and interpretability. Therefore, direct numerical comparisons between GovDocs-16, GovDocs-24, GovDocs-29, FFT-75, and private corpora should be interpreted cautiously. In this respect, our study follows the grayscale fragment representation line of research, while integrating CBAM attention into a single-stream CNN framework to improve interpretability and keep the architecture simpler than dual-branch Transformer-based models.

2.3 Attention mechanisms in vision and forensics

Attention mechanisms have become important in computer vision because they allow neural networks to emphasize the most informative features or regions of an input. Channel-level attention, such as squeeze-and-excitation networks [10], recalibrates feature channels according to their relevance. Spatial attention mechanisms, including residual attention networks [11] and differentiable visual attention approaches [12], focus on identifying important locations in feature maps.

The CBAM, presented by Woo et al. [13], combines channel attention and spatial attention in a lightweight sequential module. CBAM first determines which feature channels are important and then identifies which spatial regions should receive more attention. This design has been applied in several computer vision tasks, including image classification, object detection, and semantic segmentation [13-16]. Other attention mechanisms, such as BAM [17], sequence-to-sequence attention [18], Vision Transformers [19], and Swin Transformers [20], have also contributed to the development of attention-based learning.

In cybersecurity and digital forensics, attention mechanisms have been explored in areas such as malware detection and binary analysis [21, 22]. However, their application to file fragment classification remains less developed. For forensic fragment analysis, attention is particularly relevant because it can provide additional information about the model's decision process. Channel attention can be interpreted as emphasizing discriminative byte-value patterns, while spatial attention can highlight relevant regions within the 64 × 64 fragment representation. This makes CBAM suitable for our task, where interpretability is important in addition to classification performance. The detailed presentation of CBAM is discussed in Section 3.3.

2.4 Class imbalance challenge

Class imbalance is another important challenge in forensic datasets. In practice, some file types may be represented by a large number of source files and fragments, while other types may be rare. This issue is particularly important for fragment classification because minority classes may also correspond to compressed or composite formats, making them difficult to distinguish from other high-entropy data. As a result, overall accuracy alone may hide weak performance on minority or difficult classes.

Several strategies have been proposed to address class imbalance. SMOTE [23] generates synthetic minority samples by interpolation and has motivated several extensions. DeepSMOTE [24] combines deep feature learning with oversampling in feature space, while other studies have explored the impact of oversampling and feature engineering on imbalanced datasets [25, 26]. However, applying synthetic oversampling directly to binary fragments is not straightforward, because modifying byte sequences can alter their semantic meaning or create unrealistic fragments.

For this reason, imbalance-aware training strategies that do not alter the raw fragment content are particularly relevant to file fragment classification. In this paper, we train the model using balanced mini-batches so that each batch includes fragments from all classes. This strategy preserves the original fragments while reducing the influence of majority classes during optimization. In addition to overall accuracy, we also report balanced accuracy, macro-F1, weighted-F1, and per-class precision and recall to provide a more complete view of the model's behavior across both majority and minority classes.

3. BACKGROUND

This section highlights key information about CNNs, binary-to-grayscale conversion methodology, and the CBAM, making the paper easier to understand for both machine learning researchers and digital forensics professionals.

3.1 Convolutional Neural Networks

CNNs form a family of deep learning models that have been

specifically designed for handling structured data such as images or, in our case, 2D byte maps. There are three essential features of CNNs: local connectivity, shared weights, and hierarchical feature learning. It is because of these features that convolutional networks are especially effective for pattern recognition, including file fragment classification. Hierarchical feature learning enables the model to detect both local and structural patterns in fragment data using consecutive layers of convolution and pooling.

CNNs have been well established in their effectiveness through a succession of landmark architectures in computer vision. The success of AlexNet demonstrated the ability of deep CNNs to automatically learn hierarchical visual features from large datasets [27]. Subsequent networks such as VGGNet showed the improvement made possible by stacking deeper networks with small convolution kernels [28]. Further, ResNet provided a significant advance by adding residual learning and allowing the training of deeper networks while reducing the vanishing gradient problem [29]. DenseNet later took a further step forward by connecting all layers inside a block to allow feature reuse [30]. Finally, EfficientNet proposed a compound scaling approach that considers depth, width, and image input size simultaneously [31].

Aside from image processing, CNN-based methods have shown promise in the analysis of binary data in cybersecurity and digital forensics applications. Early studies involved efficient and automated identification of file types from binary fragments using content-based techniques [32]. Regarding malware detection and classification, CNN-based models have already succeeded in classifying binary executable files represented as images [33], indicating that CNN architectures are suitable for learning from raw binary representations. This suitability is also reflected in CNN-based malware classification techniques involving spatial attention to emphasize discriminative elements of images generated from binaries [22]. With more complex deep models becoming popular, there is a pressing need to make these models explainable in order to improve their interpretability. Explainable AI techniques enable the interpretation of how decisions are made by deep neural networks, thereby increasing transparency and trust in forensic applications [34]. There have also been advances in interpreting attention-based and transformer models, emphasizing the importance of interpretability for deep models used in security-related fields [35]. In parallel, non-deep learning techniques such as normalized compression distance has been explored in fragment classification applications [36].

3.2 Binary-to-grayscale conversion

Another common technique involves converting a sequence of binary values into a grayscale image. In our case, each 4096-byte data fragment is represented as a 64 × 64 pixel image, where bytes are mapped to pixels in the same order in which they are read from the file. Thus, each byte corresponds to one pixel, with an intensity value equal to the numerical value of that byte, ranging from 0 to 255 for grayscale values from black to white. This conversion process is illustrated in Figure 2.

In the revised experimental protocol, the first 512 bytes of each source file are skipped before fragment extraction. This choice reduces reliance on file headers and makes the task closer to headerless fragment classification. The pixel intensity values are normalized to the range [0, 1] by dividing each pixel value by 255 before being used as input to the network. This normalization step helps improve numerical stability during training by keeping activation and gradient values within manageable ranges.

The resulting image encoding shows different patterns depending on the type of file fragment. Text file fragments, which contain mostly printable ASCII characters, are displayed as fairly uniform grayscale images. Compressed or encrypted fragments tend to produce noise-like patterns due to their high entropy. Composite formats may also contain heterogeneous internal structures, making fragment-level classification more difficult when contextual information is limited.

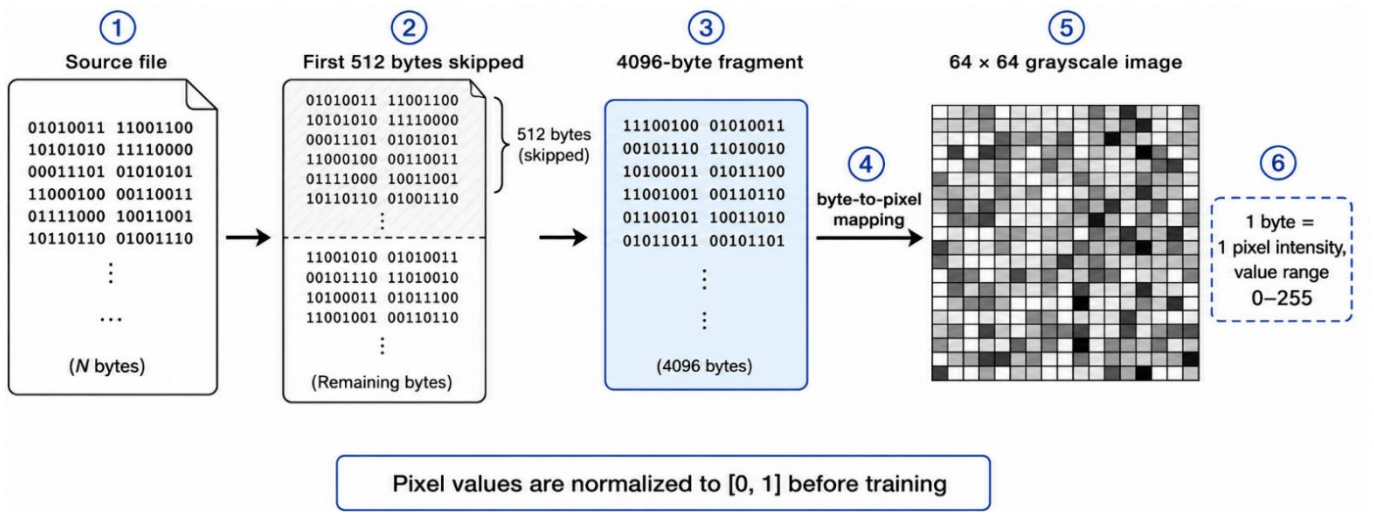

Figure 2. Binary-to-grayscale fragment conversion

3.3 Convolutional Block Attention Module

The CBAM [13] is an attention mechanism that applies both channel attention and spatial attention to the input feature map sequentially. Given an intermediate feature map F with dimensions C × H × W, where C, H, and W represent the number of channels, height, and width, respectively, CBAM first computes a 1D channel attention map Mc ∈ R^C and a 2D spatial attention map Ms ∈ R^(H × W). These attention maps are then multiplied with the input feature map F to

produce the refined feature map F′. The CBAM structure is illustrated in Figure 3.

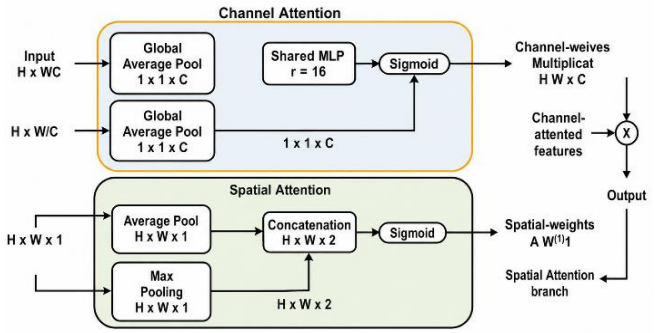

Figure 3. Convolutional Block Attention Module (CBAM) channel and spatial attention

During the channel attention stage, CBAM identifies which feature channels are more important. It utilizes global average pooling and global max pooling on the input tensor F to generate two C-dimensional feature vectors, denoted as Favg and Fmax, respectively. Each of them passes through an identical MLP with one hidden layer, with a reduction ratio r equal to 16, and the output of the MLP is:

$$Mc=\sigma(MLP(Favg)+MLP(Fmax)) \quad (1)$$

where, σ stands for the sigmoid function. *Mc* is a vector of dimension *C* whose elements lie in the range [0, 1]. Each element of Mc represents the importance of the corresponding channel. Finally, the input feature map F is scaled according to *Mc* through element-wise multiplication.

The spatial attention mechanism deals with identifying which spatial positions of the feature map are significant. For this purpose, the channel-refined feature F′, produced after the channel attention phase, is aggregated using both average pooling and max pooling with respect to the channel dimension, giving two H × W spatial feature maps. These two spatial feature maps are concatenated and convolved with a 7 × 7 kernel, followed by a sigmoid function, to obtain the final H × W spatial attention map Ms ∈ RH×W. Each value of this map is between 0 and 1 and indicates the importance of a particular spatial position across the channels.

Overall, CBAM adaptively identifies the feature channels and spatial locations within the feature maps that carry more informative content. This is achieved through the sequential application of channel and spatial attention, whereby the CNN learns what to attend to and where to focus. For our file fragment classification task, where fragments may contain compressed, encrypted, or heterogeneous byte patterns, CBAM enables the network to recalibrate its features accordingly. The proposed model uses CBAM attention maps at different intermediate feature resolutions, including 16 × 16 and 8 × 8 feature maps, which are later used to provide visual interpretation examples.

3.4 Computational efficiency and design rationale

While architectures like ByteNet have adopted dual-branch models combining Vision Transformers with CNN representations, and CNN-LSTM models have integrated recurrent components, the CNN-CBAM model takes a more efficient and interpretable approach based on the following considerations:

(1) Unified Stream Classification: In contrast to two-stream methods, where file fragment representations are classified through parallel pipelines, such as byte and image streams, the proposed model processes its input through only one stream per sample, allowing for a more compact architecture and batch classification on regular forensic computers.

(2) Lightweight Attention: The CBAM module can be integrated into the CNN backbone without introducing a separate Transformer branch or recurrent processing mechanism. Unlike global self-attention used in Transformer architectures, CBAM relies on channel reweighting and spatial convolution operations, making it suitable for efficient forensic processing.

(3) Forensic Explainability: In the context of our forensic task, channel attention reflects the importance of learned byte-value patterns, whereas spatial attention highlights discriminative regions within the 64 × 64 fragment representation. Since the revised protocol skips the first 512 bytes of each source file, these visualizations should be interpreted as fragment-level evidence rather than direct header-signature explanations.

Computational Complexity Analysis:

•Proposed CNN-CBAM: The implemented model contains 22,069,292 trainable parameters. It processes one 64 × 64 grayscale fragment through a single-stream CNN backbone with CBAM attention maps at 16 × 16 and 8 × 8 feature resolutions.

•CNN–LSTM: Uses recurrent LSTM layers after convolutional feature extraction, where sequential computation limits parallelism during inference when compared to fully convolutional networks [7].

•ByteNet (dual-branch + ViT): Has a dual-branch architecture, where one branch uses a CNN-based image representation and the other branch uses a Transformer network; therefore, ByteNet is computationally more complex than a single-stream CNN model [8].

In general, the proposed CNN-CBAM model keeps a single-stream design while providing attention-based visualizations. This makes the approach suitable for forensic triage scenarios where both classification performance and interpretability are important.

4. PROPOSED METHODOLOGY

To evaluate the proposed CNN-CBAM method, a dataset was generated from the publicly available GovDocs collection, following the recommendations of Garfinkel [37]. The dataset consists of sixteen file types selected from the GovDocs archive, covering diverse categories such as documents, images, compressed archives, and source code files. The selected classes are CSV, DOC, DOCX, GIF, GZ, HTML, JAVA, JPG, LOG, PDF, PNG, PPT, RTF, TEXT, XLS, and XML.

Each source file is first assigned to a single subset before fragment generation using a source-file-level split. This procedure is implemented to prevent fragments originating from the same source file from appearing in different subsets. After the split, each file is divided into non-overlapping 4096-byte fragments, corresponding to a typical file system cluster size. To reduce reliance on file headers and make the task closer to headerless fragment classification, the first 512 bytes of each source file are skipped before fragment extraction.

Each fragment is then reshaped into a 64 × 64 grayscale image.

The revised preparation process results in a total of 761,045 fragments across all classes. The dataset is divided into 479,946 training fragments, 53,894 validation fragments, and 227,205 held-out test fragments. This split is performed using a fixed random seed of 42, and the source-level leakage check confirms that no source file contributes fragments to more than one subset. The dataset includes high-entropy compressed-format fragments such as GZ, PNG, and DOCX, as well as composite-format fragments such as PPT and PDF. The resulting source-file and fragment counts for the training, validation, and test subsets are summarized in Table 2.

Table 2. Dataset split summary

Split	Source Files	Fragments
Train	185,181	479,946
Validation	20,576	53,894
Test	88,184	227,205
Total	293,941	761,045

4.1 Network architecture

In the proposed architecture, we employ a deep CNN with additional CBAM modules placed at selected positions within the network. The overall architecture is illustrated in Figure 4. The input to the network is a 64 × 64 grayscale representation of a 4096-byte file fragment. The initial part of the network consists of two convolutional layers with 64 filters each, followed by a max-pooling layer. This is followed by a third convolutional layer with 64 filters and a second max-pooling layer. The subsequent stage includes a convolutional layer with 64 filters, followed by another convolutional layer with 256 filters, after which a CBAM module is applied at the 16 × 16 feature-map resolution.

After this first attention refinement stage, max-pooling reduces the feature maps to an 8 × 8 resolution. A subsequent convolutional layer with 128 filters is then followed by a second CBAM block at the 8 × 8 feature-map resolution. These two CBAM stages allow the model to refine both intermediate and deeper feature representations through channel and spatial attention.

After the final attention mechanism is applied, the feature maps are flattened. The classifier component of the network consists of fully connected layers with 2048, 2048, and 256 neurons, with dropout applied after the first two 2048-neuron dense layers. Finally, the softmax output layer produces a probability distribution over the 16 file types. The implemented CNN-CBAM network contains 22,069,292 trainable parameters.

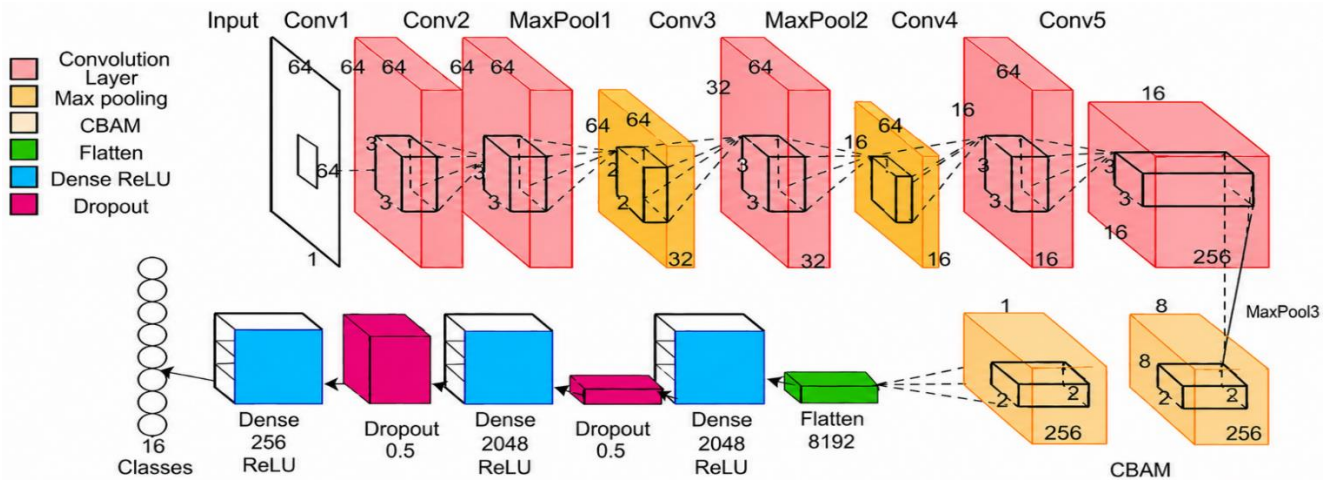

Figure 4. Proposed Convolutional Neural Network (CNN)-Convolutional Block Attention Module (CBAM) model architecture

4.2 Training configuration

The neural network implementation was carried out using Python and the TensorFlow library. Training was performed from scratch on the generated fragments. The Adam optimizer was used with an initial learning rate of 0.0005. Sparse categorical cross-entropy was used as the loss function, which is appropriate for multi-class classification. Training was conducted for up to 30 epochs with a batch size of 64. Early stopping was applied: if the validation loss did not improve for six consecutive epochs, training was stopped.

To address class imbalance among file classes, balanced mini-batches were used during training. Since the dataset contains 16 classes and the batch size is 64, each training batch contains four fragments from each class. This strategy preserves the original fragments while reducing the dominance of majority classes during optimization. No additional data augmentation was applied, since modifying binary fragments may alter their semantic structure and produce unrealistic samples.

During training, the validation set was used to monitor generalization performance and select the best model based on validation loss. The best model was identified at the epoch with the lowest validation loss and was then evaluated once on the held-out test set. The final evaluation reports accuracy, balanced accuracy, macro-F1, weighted-F1, and per-class precision, recall, and F1-score to better account for class imbalance.

4.3 Comparison with state-of-the-art methods

Direct numerical comparisons across different datasets should be made with caution, taking into account differences in fragment size, number of classes, source-file selection, train/test split protocol, and entropy characteristics. To compare our approach with other state-of-the-art methods for fragment classification, we considered several methods from the literature that use related datasets or comparable fragment representations. For completeness, classical approaches, traditional machine learning methods, and recent deep

learning models were included.

Nevertheless, such performance comparisons should not be interpreted as strictly matched benchmarks unless the same dataset subset, classes, fragment generation procedure, and split protocol are used. Therefore, the comparison presented in Table 3 should be viewed as an indicative overview of the field rather than direct evidence of absolute superiority. The closest related baseline is Chen et al. [5], which reported 70.9% accuracy on a GovDocs-based 16-class setting using 4096-byte fragments. Under our revised source-file-level protocol, the proposed CNN-CBAM model achieves 78.43% accuracy, 76.00% balanced accuracy, and 72.48% macro-F1 on 227,205 held-out test fragments.

Table 3. Indicative comparison with related methods

Method	Year	Dataset / Classes	Fragment Size	Reported Metric
Classical and Statistical Approaches				
Axelsson [38] (Compression distance)	2010	Private / 28	Various	32–36%
Veenman [39] (Statistical features)	2007	Private / 28	Various	~45%
Traditional Machine Learning on GovDocs				
Fitzgerald et al. [4] (Bag-of-bytes + SVM)	2012	GovDocs / 24	512B	47.5%
Xu et al. [40] (Grayscale image + GIST/KNN)	2014	GovDocs / 29	1024B	39.7–54.7%
Chen et al. [5] (Grayscale CNN)	2018	GovDocs / 16	4096B	70.9%
Deep Learning with Sequential or Attention Models				
Beebe et al. [2] (Sceadan, N-gram + SVM)	2013	Mixed / 38	512B	73.45%
CNN–LSTM [7]	2023	FFT-75 / 75	4096B	78.6%
ByteNet [8]	2024	FFT-75 / 75	512B	73.2%
Felemban et al. [9] (M-DSC)	2024	FFT-75 / 75	4096B	79.27%
Proposed Method				
Our CNN+CBAM	2026	GovDocs / 16	4096B	78.43% accuracy; 72.48% macro-F1

4.4 Key observations

Cautious Accuracy Comparison: The proposed model achieves an accuracy of 78.43% under a stricter source-file-level evaluation protocol. Compared with the 70.9% accuracy reported by Chen et al. [5] on a related GovDocs-based 16-class setting, this represents a positive difference of 7.53 percentage points. However, this comparison should be interpreted cautiously because the exact dataset subset, file selection, fragment generation process, and split protocol may differ.

Accuracy and Imbalance-Aware Evaluation: Since the dataset is imbalanced and some classes have smaller support values, evaluating performance using overall accuracy alone is not sufficient. Therefore, we also report balanced accuracy, macro-F1, weighted-F1, and per-class precision, recall, and F1-score. The proposed model achieves a balanced accuracy of 76.00%, a macro-F1 score of 72.48%, and a weighted-F1 score of 77.85% on the held-out test set.

Accuracy and Efficiency Balancing: The M-DSC method offers a 6× inference speed-up and four times fewer parameters than FiFTy; however, it is evaluated on FFT-75 rather than on the same GovDocs-16 setting. Therefore, the comparison should be understood mainly in terms of design philosophy. The CNN-CBAM model focuses on maintaining a single-stream architecture while providing attention-based interpretability.

Interpretability and Complexity: While the dual-branch Transformer architecture in ByteNet has higher architectural complexity because it combines both byte-level and image-based analysis, CNN-CBAM uses a single-stream CNN that incorporates channel and spatial attention maps. In forensic applications, the ability to identify which learned byte patterns and fragment regions influence the classification decision can be as important as the final accuracy value.

5. EXPERIMENT RESULTS

The trained CNN-CBAM model was evaluated using the held-out test dataset, which comprised 227,205 fragments covering all 16 file types. Under the revised source-file-level split, the model achieved an overall accuracy of 78.43%. Since the dataset is imbalanced, accuracy alone is not sufficient to fully evaluate performance. Therefore, we also report balanced accuracy, macro-F1, weighted-F1, and per-class precision, recall, and F1-score. The model achieved a balanced accuracy of 76.00%, a macro-F1 score of 72.48%, and a weighted-F1 score of 77.85%. These values provide a more complete view of the model's behavior across both majority and minority classes.

5.1 Confusion matrix analysis

From the normalized confusion matrix shown in Figure 5, one can observe that some file types exhibit high classification performance. In particular, CSV, GIF, LOG, XLS, and XML demonstrate high recall values, indicated by strong diagonal values in the normalized matrix. CSV reaches a recall of 95.83%, GIF reaches 94.31%, LOG reaches 96.39%, XLS reaches 91.86%, and XML reaches 92.31%. These classes contain byte-level patterns that are more distinguishable than many compressed or composite formats.

HTML also shows good separability, with a recall of 86.47%, while JAVA achieves a high recall of 94.61% despite its limited support. This suggests that source-code-like byte structures can remain distinguishable even when the number of available fragments is relatively small. However, the lower precision of JAVA indicates that some fragments from other classes are incorrectly classified as JAVA, which should be interpreted carefully.

However, there are also certain classes that the model finds more difficult to classify. These include DOCX, PNG, PPT, PDF, and DOC fragments, which obtain lower F1-scores compared with more distinctive text-oriented or structured file types. The difficulty encountered with these file types is mainly related to compression, composite internal structures, high-entropy byte distributions, limited source-file diversity, and limited fragment-level context. DOCX refers to Microsoft Word Open XML, which is a ZIP-based compressed format.

Similarly, PNG fragments display noise-like byte distributions due to compression. Therefore, when only a 4096-byte fragment is available and the header region is skipped, these classes may provide limited discriminative information.

In general, the error cases away from the diagonal indicate that mistakes are made between classes that are either similar in their content characteristics or where one of the classes has relatively fewer instances. Examples include confusion among compressed or composite formats, such as DOCX, PNG, PDF, and PPT. These errors reflect the intrinsic difficulty of distinguishing high-entropy or container-based fragments without additional contextual information.

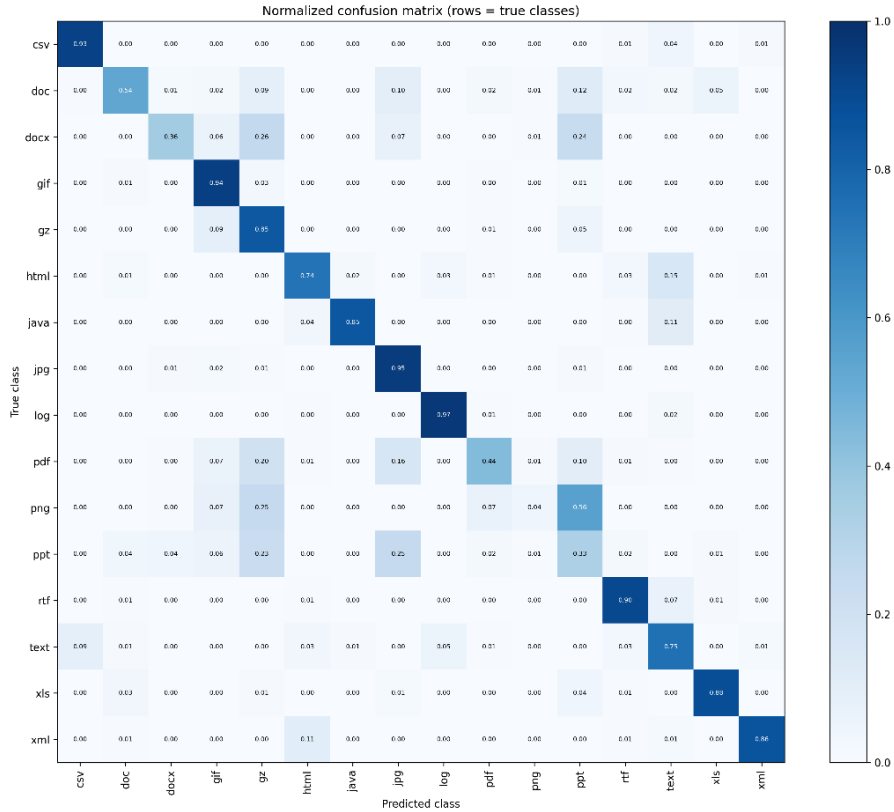

Figure 5. Row-normalized confusion matrix of the proposed Convolutional Neural Network (CNN)-Convolutional Block Attention Module (CBAM) model

5.2 Confusion matrix

The row-normalized confusion matrix of the proposed CNN-CBAM model over the 16 file types is shown in Figure 5. The rows represent the ground-truth classes, whereas the columns represent the predicted classes. The matrix is normalized by true class, so each row represents the distribution of predictions for one actual class. The diagonal entries show the correctly classified fragments for each class.

As shown in Figure 5, the CNN-CBAM classifier performs strongly on several classes, including CSV, GIF, LOG, XLS, and XML. In contrast, compressed and composite formats such as DOCX, PNG, PDF, and PPT show higher confusion due to their high-entropy byte distributions and internal container structures. DOC and PDF also remain challenging because fragments from these formats may contain heterogeneous internal content, making their local byte patterns less consistent across files. The row-normalized matrix highlights per-class behavior independently of class support, making it easier to identify classes with high recall and classes affected by confusion with compressed or composite formats.

5.3 Per-class performance

For the per-class performance analysis, Table 4 reports precision, recall, and F1-score values for each file class. The strongest results are obtained for LOG, CSV, XML, GIF, and XLS. LOG achieves an F1-score of 0.9528, followed by CSV with 0.9311, XML with 0.9313, GIF with 0.9142, and XLS with 0.9035. These results confirm that the model performs well on classes with more distinctive byte distributions or more consistent structural patterns.

Table 4. Per-class test-set metrics

File Type	Precision	Recall	F1-Score	Support
CSV	0.9054	0.9583	0.9311	18,900
DOC	0.8392	0.5514	0.6655	18,900
DOCX	0.2378	0.4107	0.3012	655
GIF	0.8871	0.9431	0.9142	18,900
GZ	0.5728	0.8793	0.6937	18,900
HTML	0.8737	0.8647	0.8692	18,900
JAVA	0.5408	0.9461	0.6882	427
JPG	0.6759	0.9098	0.7756	18,900
LOG	0.9420	0.9639	0.9528	18,900
PDF	0.8124	0.4801	0.6035	18,900
PNG	0.2256	0.4069	0.2902	2,802
PPT	0.5580	0.3531	0.4325	18,900
RTF	0.7922	0.8866	0.8368	3,871
TEXT	0.8546	0.7650	0.8073	18,900
XLS	0.8888	0.9186	0.9035	18,900
XML	0.9395	0.9231	0.9313	11,550

Other classes obtain moderate performance. HTML reaches an F1-score of 0.8692, RTF reaches 0.8368, TEXT reaches 0.8073, and JPG reaches 0.7756. GZ obtains a high recall of 0.8793 but a lower precision of 0.5728, which indicates that the model often recognizes GZ fragments but also confuses some fragments from other classes with GZ.

However, compressed and composite file formats such as DOCX, PNG, PDF, and PPT exhibit relatively lower F1-scores. For instance, DOCX achieves an F1-score of 0.3012, PNG achieves 0.2902, PPT achieves 0.4325, and PDF achieves 0.6035. This is because DOCX and PNG fragments contain compressed data, making them high-entropy byte sequences by nature and therefore difficult to distinguish from other compressed or composite fragments. In addition, the model may confuse DOCX, PDF, and PPT fragments because Office and document formats can contain heterogeneous internal structures and compressed components.

Importantly, while our approach incorporates attention-based mechanisms in order to detect fine-grained features, the inherent problem associated with high-entropy content still exists. When byte sequences are close to uniformly random, file-type classification becomes extremely difficult without additional contextual information. This is reflected in the obtained results, where the lowest F1-scores are observed for PNG and DOCX, followed by PPT and PDF.

Overall, the model performs strongly across several file types and shows improved imbalance-aware performance compared with the previous training protocol. Nonetheless, some classes still remain challenging due to constraints related to compressed content, composite internal structures, limited source-file diversity, and limited fragment-level context.

5.4 Attention map visualization

As illustrated in Figure 6, the CBAM attention maps are visualized for both correctly classified fragments and error cases. These examples show how the model focuses on different regions of the 64 × 64 fragment representation. The visualizations should be interpreted as indicators of model behavior rather than complete causal explanations.

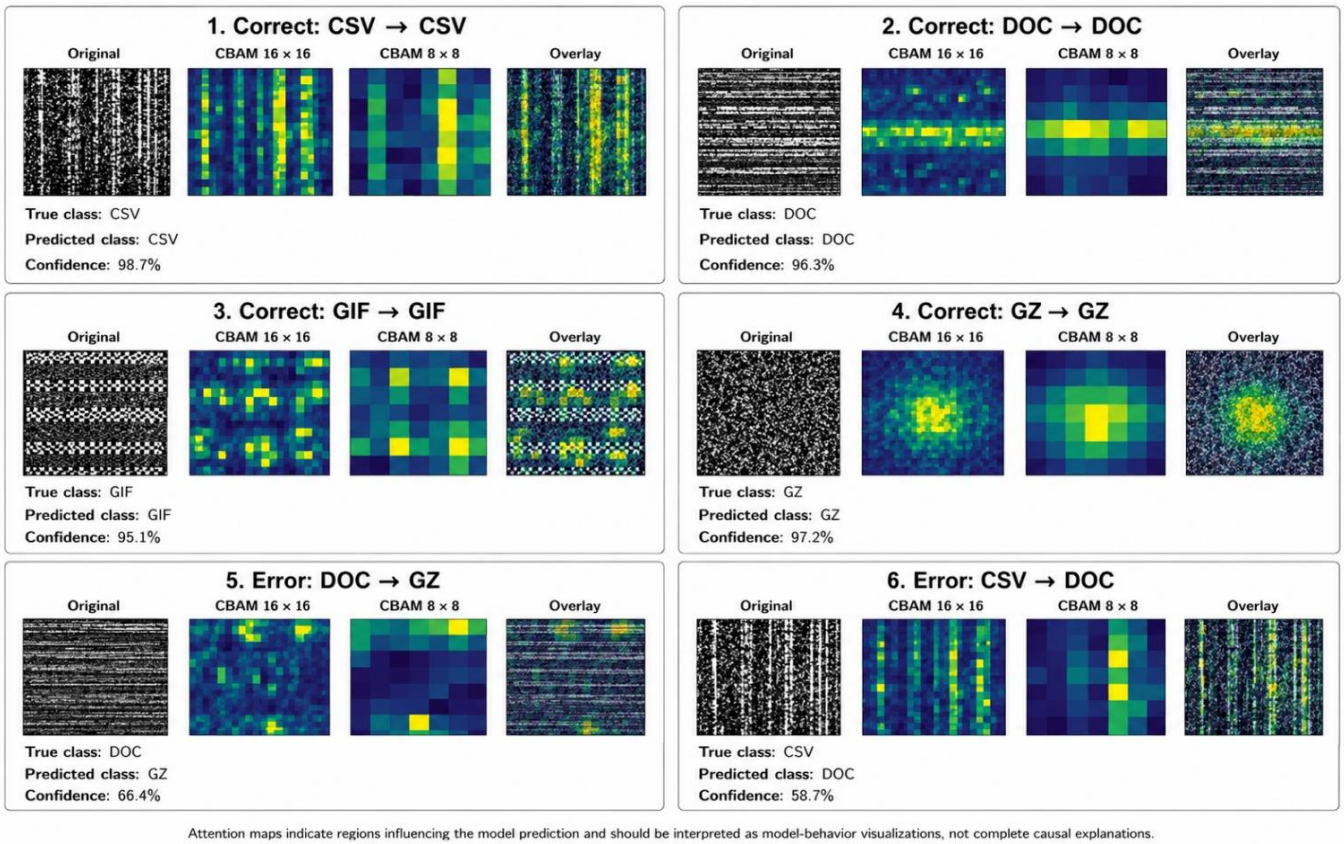

Figure 6. Convolutional Block Attention Module (CBAM) attention maps for correct and error cases

6. DISCUSSION

6.1 Advantages and contributions

The results indicate that integrating attention layers such as CBAM into the CNN architecture is beneficial for file fragment classification. Under the revised source-file-level evaluation protocol, the CNN-CBAM model achieved 78.43% accuracy, 76.00% balanced accuracy, 72.48% macro-F1, and 77.85% weighted-F1 on 227,205 held-out test fragments. Although these results are lower than those obtained under the previous fragment-level evaluation protocol, they provide a more reliable performance estimate because fragments originating from the same source file are prevented from appearing in different subsets.

This indicates that attention layers can help the model learn more discriminative features in cases where fragment categories exhibit strong statistical similarities. For example, channel attention allows the network to emphasize learned byte-value patterns, whereas spatial attention assists in recognizing relevant local structures within the 64 × 64 fragment representation. This is particularly useful for fragment classification, where the model must operate without relying on filenames, extensions, or header information.

Another significant insight of the proposed work is that the combination of channel and spatial attention provides a meaningful way to analyze the model's internal behavior. Channel attention focuses on what types of learned features are important, while spatial attention helps identify where discriminative regions are located within the fragment representation. In this regard, CBAM is not only used as a performance-oriented module, but also as an interpretability mechanism that can support forensic analysis.

Another advantage of our approach is its improved interpretability. By analyzing the attention maps generated by CBAM, the forensic examiner can obtain preliminary insight into why a particular fragment was classified as belonging to a specific file type. This is illustrated by the attention-map examples, which show how the model focuses on different spatial regions for correctly classified fragments and error cases. These visualizations do not replace expert forensic interpretation; however, they provide useful evidence regarding the regions that influenced the model's classification decision.

Computationally, the model preserves a single-stream CNN structure and does not require a dual-branch Transformer architecture or recurrent processing. The implemented CNN-CBAM model contains 22,069,292 trainable parameters and processes each fragment as a 64 × 64 grayscale image. This makes the approach practically feasible for forensic triage scenarios where large numbers of fragments may need to be analyzed.

6.2 Limitations and challenges

Despite these improvements, certain limitations remain. Some file types with limited source-file diversity or limited fragment support, such as DOCX, JAVA, PNG, and RTF, remain difficult to classify. Although JAVA achieves high recall in the revised experiment, its lower precision indicates that some fragments from other classes are incorrectly assigned to this class. DOCX and PNG remain among the most challenging classes, mainly because their compressed internal structures can produce high-entropy byte distributions that are difficult to distinguish at the fragment level.

One of the challenges that emerges from the experiments is the classification of highly compressed or information-poor data segments. Despite the presence of attention mechanisms, identifying highly entropic fragments remains difficult. This is evident for PNG and DOCX fragments, which may have noise-like appearances and may therefore be confused with other compressed or composite formats. The problem becomes more apparent when the fragment itself contains little discriminative signal, especially after the header region has been skipped.

Another limitation is related to composite file formats such as PDF, PPT, and DOC. These file types may contain heterogeneous internal structures, embedded objects, compressed components, or text-like regions depending on the specific source file and fragment offset. As a result, fragments from these classes may not always present consistent byte-level patterns, which increases confusion with other document or archive-like formats.

A further limitation associated with the proposed research is that we make the closed-set assumption, where the model is aware of the set of file types and assumes that each fragment belongs to one of the 16 classes. However, in practice, a fragment may come from a file type that was unknown to the system during training. As a result, the system would assign the fragment to one of the known classes, which may become an issue in real forensic investigations.

Finally, despite the benefits of attention-based interpretability, attention maps should be interpreted with caution. Although they highlight regions that influence the model's internal activations, they do not provide a complete explanation of the decision-making process. Manual analysis of attention maps for each fragment may also be time-consuming in large-scale forensic cases. Therefore, further development of automatic interpretation and summarization methods may be considered.

6.3 Future research directions

Future research directions include the following.

Hybrid Architectures for Next-Generation Forensics: Possible future research may involve exploring new hybrid architectures by combining modern architectural advances with the interpretability offered by CBAM to further improve file fragment classification. One such direction involves incorporating CBAM with intrabyte analysis methods. For example, using CBAM together with the 1-bit sliding window mechanism proposed in ByteNet [8] could allow the model to capture fine-grained patterns at the bit level inside each byte while retaining spatial interpretability. Such an architecture may be useful especially in cases where variable-length encoding or compressed representations are used.

Another possible approach is the use of efficient variations of Transformers. The application of Linear Attention or Performer-based architectures may reduce the quadratic complexity of regular self-attention while preserving the capacity to capture long-range relations. Nevertheless, the combination of CBAM with any Transformer should take into consideration the need to preserve forensic interpretability and avoid excessive architectural complexity.

Dynamic attention routing can also be considered for enhancing computational efficiency and classification robustness. Learnable dynamic routing could selectively use or skip some attention components based on the characteristics of the fragment. For instance, high-entropy fragments may require different attention behavior from text-oriented fragments, because they contain fewer obvious byte-value regularities.

Second, implementing unknown file type recognition or open-set classification represents a useful avenue for development. In a real-world setting, a fragment may not belong to any of the known types used during training. Methods such as outlier detection, uncertainty estimation, or open-set recognition could allow the algorithm to identify fragments of an unknown file type rather than incorrectly categorizing them as one of the known types.

Third, few-shot learning techniques may help address the problem of rare file formats. While traditional classification methods require numerous examples of a new file format for training classifiers, few-shot learning techniques seek to adapt a model to a new class using very few examples. In this case, either Prototypical Networks or Model-Agnostic Meta-Learning (MAML) could be used to rapidly learn new fragment categories.

Standardized Cross-Dataset Benchmarking: There is an immediate need for standard evaluation methodologies that allow fair comparisons between different techniques. Another important future direction is evaluating model generalization

using cross-dataset tests. Our suggestions include:

•Transfer Learning Evaluation: Evaluating the ability of a model trained using FFT-75 to be tested on GovDocs, or the opposite scenario.

•Efficiency Metrics: Reporting computational requirements such as FLOPs, time required to process a single fragment, peak memory usage, and number of parameters in addition to accuracy scores.

•Open-Source Interpretability Tools: Developing attention visualization software tailored to forensic applications, allowing non-experts in deep learning to review and interpret the decisions made by the model.

•Adversarial Robustness Testing: Checking the behavior of models when processing fragments that have been deliberately obfuscated in anti-forensic ways.

It is important for practical use that fragment classifiers perform effectively across corpora with different distributions. Finally, our future work includes expanding interpretability by looking into methods beyond attention visualization. Although CBAM helps us understand some aspects of the model's decisions, explainable artificial intelligence techniques may provide further clarity in forensic analysis.

7. CONCLUSION

File fragment identification is one of the fundamental tasks in digital forensics, as it enables fragments to be classified based on their content when metadata-based techniques fail due to corruption or anti-forensic methods. In this study, we implemented a CNN architecture integrating the CBAM module to classify sixteen file types from the GovDocs dataset. Under the revised source-file-level evaluation protocol, the proposed model achieved an accuracy of 78.43%, a balanced accuracy of 76.00%, a macro-F1 score of 72.48%, and a weighted-F1 score of 77.85% on 227,205 held-out test fragments.

Compared with the 70.9% accuracy reported by Chen et al. [5] on a related GovDocs-based 16-class setting, the achieved result remains competitive. Nevertheless, this comparison should be interpreted cautiously because the exact file selection, fragment generation process, and split protocol may differ from ours. The main advantage of the revised evaluation protocol is that fragments from the same original source file are prevented from appearing in different subsets, providing a stricter and more reliable estimate of performance.

Contrary to modern Transformer models such as ByteNet, which rely on more complex architectures through dual-branch processing, or very lightweight models such as M-DSC, which prioritize inference speed and model compactness, our model delivers a balanced solution that fulfills three essential criteria for forensic applications:

(1) Accurate and imbalance-aware: It achieves competitive performance on GovDocs-16 while reporting not only accuracy, but also balanced accuracy, macro-F1, weighted-F1, and per-class metrics.

(2) Forensically Interpretable: Channel attention emphasizes learned byte-value patterns, whereas spatial attention identifies discriminative regions within the 64 × 64 fragment representation.

(3) Practically Feasible: The model supports forensic triage through single-stream processing without requiring a dual-branch Transformer architecture or recurrent processing.

The main value of this research comes from showing that attention mechanisms developed in 2018, namely CBAM, can still demonstrate useful performance and interpretability when they are applied to CNN-based file fragment classification. As opposed to many more sophisticated models created in recent years, CBAM offers a clear attention breakdown based on two levels: the feature level, which indicates what is important, and the location level, which indicates where discriminative evidence may be found. This structure is well aligned with forensic investigation needs.

The revised results also show that some challenges remain. High-entropy and composite formats such as DOCX, PNG, PDF, and PPT continue to be difficult to classify at the fragment level, especially when the available fragment contains limited discriminative information. Although balanced mini-batch training improves imbalance-aware performance, it does not fully eliminate the intrinsic difficulty of distinguishing compressed or container-based formats from short headerless fragments.

Although progress has been made, difficulties still persist, especially when dealing with fragments that are intrinsically information-poor, highly compressed, or associated with limited source-file diversity. In light of these results, there is a need for more research focusing on unknown file type identification, cross-dataset evaluation, and learning from limited training data. Methods such as open-set recognition, few-shot learning, and hybrid models that combine attention-based interpretability with intrabyte analysis could prove helpful in this area.

This paper shows that a carefully designed attention-based CNN can achieve competitive performance while preserving the interpretability and computational feasibility needed for practical forensic applications. Further investigation into hybrid systems that combine the interpretability offered by CBAM with recent advancements in intrabyte analysis and efficient Transformers could be highly productive. Such approaches will be important in light of the increasing complexity of obfuscation techniques used by potential adversaries.

REFERENCES

[1] McDaniel, M., Heydari, M.H. (2003). Content based file type detection algorithms. In Proceedings of the 2003 36th Annual Hawaii International Conference on System Sciences, Big Island, USA, p. 10. https://doi.org/10.1109/HICSS.2003.1174905

[2] Beebe, N.L., Maddox, L.A., Liu, L., Sun, M. (2013). Sceadan: Using concatenated n-gram vectors for improved file and data type classification. IEEE Transactions on Information Forensics and Security, 8(9): 1519-1530. https://doi.org/10.1109/TIFS.2013.2274728

[3] Amirani, M.C., Toorani, M., Beheshti, A. (2008). A new approach to content-based file type detection. In 2008 IEEE Symposium on Computers and Communications, Marrakech, Morocco, pp. 1103-1108. https://doi.org/10.1109/ISCC.2008.4625611

[4] Fitzgerald, S., Mathews, G., Morris, C., Zhulyn, O. (2012). Using NLP techniques for file fragment classification. Digital Investigation, 9: S44-S49. https://doi.org/10.1016/j.diin.2012.05.008

[5] Chen, Q., Liao, Q., Jiang, Z.L., et al. (2018). File fragment classification using grayscale image conversion

and deep learning in digital forensics. In 2018 IEEE Security and Privacy Workshops (SPW), San Francisco, USA, pp. 140-147. https://doi.org/10.1109/SPW.2018.00029

[6] Wang, Y., Su, Z., Song, D. (2018). File fragment type identification with convolutional neural networks. In Proceedings of the 2018 International Conference on Machine Learning Technologies, Jinan, China, pp. 41-47. https://doi.org/10.1145/3231884.3231889

[7] Zhu, N., Liu, Y., Wang, K., Ma, C. (2023). File fragment type identification based on CNN and LSTM. In Proceedings of the 2023 7th International Conference on Digital Signal Processing, Chengdu, China, pp. 16-22. https://doi.org/10.1145/3585542.3585545

[8] Liu, W., Wu, K., Liu, T., Wang, Y., Yap, K.H., Chau, L.P. (2024). ByteNet: Rethinking multimedia file fragment classification through visual perspectives. IEEE Transactions on Multimedia, 27: 1305-1319. https://doi.org/10.1109/TMM.2024.3521830

[9] Felemban, M., Ghaleb, M., Saaim, K., Alsaleh, S., Almulhem, A. (2024). File fragment type classification using light-weight convolutional neural networks. IEEE Access, 12: 157179-157191. https://doi.org/10.1109/ACCESS.2024.3486180

[10] Hu, J., Shen, L., Albanie, S., Sun, G., Wu, E. (2020). Squeeze-and-excitation networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 42(8): 2011-2023. https://doi.org/10.1109/tpami.2019.2913372

[11] Wang, F., Jiang, M., Qian, C., et al. (2017). Residual attention network for image classification. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, USA, pp. 6450-6458. https://doi.org/10.1109/CVPR.2017.683

[12] Jetley, S., Lord, N.A., Lee, N., Torr, P.H. (2018). Learn to pay attention. arXiv preprint arXiv:1804.02391. https://doi.org/10.48550/arXiv.1804.02391

[13] Woo, S., Park, J., Lee, J.Y., Kweon, I.S. (2018). CBAM: Convolutional block attention module. In European Conference on Computer Vision, pp. 3-19. https://doi.org/10.1007/978-3-030-01234-2_1

[14] Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, USA, pp. 248-255. https://doi.org/10.1109/CVPR.2009.5206848

[15] Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P. (2020). Focal loss for dense object detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 42(2): 318-327. https://doi.org/10.1109/TPAMI.2018.2858826

[16] Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. In European Conference on Computer Vision, pp. 833-851. https://doi.org/10.1007/978-3-030-01234-2_49

[17] Park, J., Woo, S., Lee, J.Y., Kweon, I.S. (2018). Bam: Bottleneck attention module. arXiv preprint arXiv:1807.06514. https://doi.org/10.48550/arXiv.1807.06514

[18] Bahdanau, D., Cho, K., Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473. https://doi.org/10.48550/arXiv.1409.0473

[19] Dosovitskiy, A., Beyer, L., Kolesnikov, A., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929. https://doi.org/10.48550/arXiv.2010.11929

[20] Liu, Z., Lin, Y., Cao, Y., et al. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, Canada, pp. 9992-10002. https://doi.org/10.1109/ICCV48922.2021.00986

[21] Liu, L., Wang, B.S., Yu, B., Zhong, Q.X. (2017). Automatic malware classification and new malware detection using machine learning. Frontiers of Information Technology & Electronic Engineering, 18(9): 1336-1347. https://doi.org/10.1631/FITEE.1601325

[22] Yakura, H., Shinozaki, S., Nishimura, R., Oyama, Y., Sakuma, J. (2018). Malware analysis of imaged binary samples by convolutional neural network with attention mechanism. In Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy, Tempe, USA, pp. 127-134. https://doi.org/10.1145/3176258.3176335

[23] Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P. (2002). SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16: 321-357. https://doi.org/10.1613/jair.953

[24] Dablain, D., Krawczyk, B., Chawla, N.V. (2022). DeepSMOTE: Fusing deep learning and SMOTE for imbalanced data. IEEE Transactions on Neural Networks and Learning Systems, 34(9): 6390-6404. https://doi.org/10.1109/TNNLS.2021.3136503

[25] Blagus, R., Lusa, L. (2013). SMOTE for high-dimensional class-imbalanced data. BMC Bioinformatics, 14(1): 106. https://doi.org/10.1186/1471-2105-14-106

[26] Mujahid, M., Kına, E.R.O.L., Rustam, F., et al. (2024). Data oversampling and imbalanced datasets: An investigation of performance for machine learning and feature engineering. Journal of Big Data, 11(1): 87. https://doi.org/10.1186/s40537-024-00943-4

[27] Krizhevsky, A., Sutskever, I., Hinton, G.E. (2017). ImageNet classification with deep convolutional neural networks. Communications of the ACM, 60(6): 84-90. https://doi.org/10.1145/3065386

[28] Simonyan, K., Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556. https://doi.org/10.48550/arXiv.1409.1556

[29] He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep residual learning for image recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA. https://doi.org/10.1109/CVPR.2016.90

[30] Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q. (2017). Densely connected convolutional networks. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, USA, pp. 2261-2269. https://doi.org/10.1109/CVPR.2017.243

[31] Tan, M., Le, Q.V. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. In Proceedings of the 36th International Conference on Machine Learning, PMLR, pp. 6105-6114. https://proceedings.mlr.press/v97/tan19a.html?ref=ji.

[32] Penrose, P., Macfarlane, R., Buchanan, W.J. (2013). Approaches to the classification of high entropy file

fragments. Digital Investigation, 10(4): 372-384. https://doi.org/10.1016/j.diin.2013.08.004
[33] Saxe, J., Berlin, K. (2015). Deep neural network based malware detection using two dimensional binary program features. In 2015 10th International Conference on Malicious and Unwanted Software (MALWARE), Fajardo, USA, pp. 11-20. https://doi.org/10.1109/MALWARE.2015.7413680
[34] Samek, W., Wiegand, T., Müller, K.R. (2017). Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. arXiv preprint arXiv:1708.08296. https://doi.org/10.48550/arXiv.1708.08296
[35] Chefer, H., Gur, S., Wolf, L. (2021). Transformer interpretability beyond attention visualization. In 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, USA, pp. 782-791. https://doi.org/10.1109/CVPR46437.2021.00084
[36] Axelsson, S. (2010). Using normalized compression distance for classifying file fragments. In 2010 International Conference on Availability, Reliability and Security, Krakow, Poland, pp. 641-646. https://doi.org/10.1109/ARES.2010.100
[37] Garfinkel, S., Farrell, P., Roussev, V., Dinolt, G. (2009). Bringing science to digital forensics with standardized forensic corpora. Digital Investigation, 6: S2-S11. https://doi.org/10.1016/j.diin.2009.06.016
[38] Axelsson, S. (2010). The normalised compression distance as a file fragment classifier. Digital Investigation, 7: S24-S31. https://doi.org/10.1016/j.diin.2010.05.004
[39] Veenman, C.J. (2007). Statistical disk cluster classification for file carving. In Third International Symposium on Information Assurance and Security, Manchester, UK, pp. 393-398. https://doi.org/10.1109/IAS.2007.75
[40] Xu, T., Xu, M., Ren, Y., Xu, J., Zhang, H., Zheng, N. (2014). A file fragment classification method based on grayscale image. Journal of Computers, 9(8): 1863-1870. https://doi.org/10.4304/jcp.9.8.1863-1870

NOMENCLATURE

F	Intermediate feature map
C	Number of channels
H	Height of the feature map
W	Width of the feature map
Mc	Channel attention map
Ms	Spatial attention map
Favg	Average-pooled feature descriptor
Fmax	Max-pooled feature descriptor
σ	Sigmoid activation function