

Received 5 February 2023, accepted 23 February 2023, date of publication 6 March 2023, date of current version 10 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3253770

## RESEARCH ARTICLE

# Attention-Based Cross-Modal CNN Using Non-Disassembled Files for Malware Classification

JEONGWOO KIM<sup>1</sup>, JOON-YOUNG PAIK<sup>2</sup>, AND EUN-SUN CHO<sup>1</sup>, (Member, IEEE)

<sup>1</sup>Department of Computer Science and Engineering, Chungnam National University, Daejeon 34134, South Korea

<sup>2</sup>School of Software, Tiangong University, Tianjin 300387, China

Corresponding authors: Joon-Young Paik (pky2018@tiangong.edu.cn) and Eun-Sun Cho (eschough@cnu.ac.kr)

This work was supported by the research fund of Chungnam National University.

**ABSTRACT** The role of malware classification is crucial in addressing the explosive increase in malware variants. By classifying malware instances into malware families, malware analysts can apply appropriate techniques and tools to handle malware variants in each family. Using high-level representations of malware, such as disassembled codes, yields meaningful classification performance. However, malware classification based on disassembled codes depends on the practically implausible assumption that every malware is correctly reversed by disassemblers. Unfortunately, sophisticated malware, which has anti-disassembly capabilities, seeks to confuse disassemblers, yielding incorrectly disassembled codes. In this study, we focus on malware family classification, which requires no disassembly, and propose a new CNN-based malware classification model using non-disassembled malware files (i.e., binary files). Our model associates two modalities: “malware images” and “structural entropies,” which are converted and extracted from binary files. Both modalities have different granularities of bytes and chunks that complement each other. The model adopts a cross-modal attention mechanism to combine the features of the two modalities by moderating their expressive limitations. We validate our model using three popular datasets from the Kaggle Microsoft Malware Classification, Maling, and BODMAS datasets. The experimental results show that our model identifies malware families with a higher degree of accuracy than previous methods and does not require the burden of disassembling.

**INDEX TERMS** Malware classification, structural entropy, malware image, deep learning, convolutional neural network, attention mechanism.

## I. INTRODUCTION

The recent increase in remote education and work-from-home due to the COVID-19 pandemic has introduced ever-increasing threats to cybersecurity. Social phishing attacks on new vaccines, government policies, and online meeting schedules can easily deceive the workforce and students. Online collaboration platforms can be abused for the delivery of malware. Moreover, malware has continued to evolve rapidly. The logic of malware has become more sophisticated; malicious behaviors exhibited

by malware samples have increased from 9 to 12 on average during the pandemic [1]. New business models, such as malware-as-a-service, provide illegal services for pay-for-use malware, making it easier to launch malware.

With the rapid development of malware, the role of malware family classification has become essential. A malware family classifier classifies malware samples into malware families by exploiting the shared code fragments or common behavioral patterns, or attack strategies of each family. This reduces the complexity of the analysis because malware analysts heuristically know how to dissect malware samples of each family. In other words, accurate malware

The associate editor coordinating the review of this manuscript and approving it for publication was Yudong Zhang<sup>1</sup>.

family classification is indispensable for promptly dealing with sharp rise in variants of malware families [2], [3].

Deep-learning-based malware family classification has attracted considerable attention, motivated by the significant success of deep-learning techniques in the fields of vision and natural language processing. Malware family classification models based on deep neural networks outperform conventional machine-learning-based classification models. The former can learn different levels of features without any expert intervention, whereas the latter depends on handcrafted features selected based on experts' experience. Owing to their ability to learn automatically, various features can be used as inputs for deep-learning-based models, broadly categorized into dynamic and static features.

Dynamic features [4], [5], [6] refer to the runtime behavior of malware. These features (e.g., a call sequence of application programming interfaces (APIs), network behavior, memory usage, registry changes, and execution paths) are extracted by executing malware in a controlled environment (e.g., a virtual environment). They have the advantage of transparently revealing their maliciousness. However, dynamic features involve executing malware, which is time-consuming because the period of execution time is guaranteed to log useful runtime information. Building an environment that satisfies the execution conditions for malware is burdensome. Moreover, sophisticated malware adopts anti-VM, anti-debugging, and anti-emulation techniques. When they identify the existence of a VM, debugger, or emulator, they stop or behave similar to benign software.

Static features are directly extracted from binary or disassembled files without executing malware [2], [7], [8]. First, the static features of the binaries are byte sequences, byte entropy, histogram, structural entropy, and printable strings. They were all imprinted with raw bytes of binaries. Thus, this type of static feature has high applicability because there are no additional steps (i.e., execution and disassembly). Second, the disassembled files provide semantic information. Static features revealed in disassembled files include opcodes, control flow graphs, and function call graphs. Their combination can represent malicious intent. However, these features largely rely on the capabilities of a disassembler such as IDA Pro [9]. In most case, Disassembly is not guaranteed to be correct. The static features captured from incorrectly disassembled codes would contain false semantic information. Even classification models trained on such untrustworthy disassembled codes would degrade their performance. In particular, disassembling malware accurately is challenging because of the use of anti-disassembly techniques (i.e., packing and obfuscation) that encode, compress, and encrypt code and malware data. Therefore, the use of static features in disassembled codes is limited to defending against sophisticated malware families.

Multi-modal learning creates rich feature representations by combining information from various modalities (e.g.,

images and texts). Complementary information on different modalities enables a holistic exploration of feature space. Multi-modal learning is a promising solution for malware family classification because static and dynamic features of malware can be represented in diverse modalities (e.g., images and sequences of opcodes). HYDRA [10] uses static features extracted from disassembled codes and binary files. These are the APIs, opcodes, and raw bytes. HYDRA has three components to handle these different types of inputs. Orthrur [11] is a bimodal-based malware classification model. It employs two CNNs using two different modalities as input: a byte sequence and an opcode sequence. In [12], Efficient-Net processed malware images visualized from disassembled files, and 1D-CNN dealt with a byte sequence. Different CNNs were employed, considering the nature of different modalities. These multi-modal-based classification models have proven effective for malware family classification because multi-modal learning enhances malware family models by simultaneously fusing data from diverse modalities. However, when multi-modal-based malware family classification models leverage features extracted from disassembled codes, the classification models inherit the limitations mentioned above. Therefore, multi-modal models using the static features of disassembled codes would also have difficulty accurately identifying malware that can deceive disassemblers.

In this paper, we propose a novel attention-based cross-modal model for malware family classification. The goal is to identify malware families by comprehensively exploiting the static features of diverse modalities in a non-disassembled file (i.e., a binary file). Our focus on non-disassembled files allows the proposed model to process malware without considering disassembly issues. Our multi-modal approach uses malware images and the structural entropy of malware. The former is visualized from binaries with byte granularity, whereas the latter is calculated from binaries with chunk granularity (i.e., 512 bytes, 1024 bytes). Each modality has proven to be effective in various malware analyses [13], [14], [15].

Moreover, the use of both can compensate for each other; malware images are sensitive to byte distortions of code obfuscation techniques, while structural entropy is robust to byte-level modification because of its chunk-level calculation [15]. In contrast, structural entropy loses detailed information about byte-level patterns, while malware images retain byte-level information in pixels. To maximize the synergy effect through a combination of malware images and structural entropy, we devised a new attention-based cross-modal convolutional neural network (CNN) [16]. Two independent CNNs for images and structural entropy were employed to yield the respective intermediate representations. The two representations are fused by the cross-modal attention module, where they attend to each other to explore the correspondence between the images and structural entropy. We compared our model with previous classification

models to verify its effectiveness and observed performance improvements.

The main contributions of our research are as follows;

- We noticed that the different granularities (i.e., bytes and chunks) of malware images and structural entropy could compensate for each other's drawbacks. They were created from binaries without disassembly, rendering our model accurate for classifying malware that the disassembler cannot correctly reverse.
- Our cross-modal attention mechanism was devised to fuse the multi-modal features better. This fusion mechanism allows our model to learn the alignment between the byte-level and chunk-level features from the two modalities, thereby creating a representation that contains various levels of information.
- We demonstrate the effectiveness of our malware family classification model using three malware datasets. We compared it with unimodal and multi-modal models that use different fusion mechanisms. In addition, our model outperforms previous malware classification models that use other static features in malware binaries.

The remainder of this paper is organized as follows; Section II describes deep-learning-based malware classification research and related background knowledge. In Sections III to VII, we explain each component of the proposed classification model based on a cross-modal attention mechanism. Section VIII presents the performance of the proposed malware classification model on the Kaggle Microsoft malware classification dataset [17], Malimg [18], and BODMAS [19]. We compared our model with other classification models. Finally, we conclude the paper in Section IX.

## II. RELATED WORK AND BACKGROUND

This section provides previous related works on deep-learning-based malware analysis, including malware classification, and the background of our work. The first two subsections present previous studies on the use of static features with and without disassembly. Then, we discuss malware classification models with dynamic features in Section II-C. Section II-D. describes how multi-modal learning for multiple modalities has been used for malware classification.

### A. MALWARE CLASSIFICATION USING STATIC FEATURES WITH DISASSEMBLY

Many previous malware classification models depend on features extracted from disassembled binary files. One of the main features observed in the disassembled files is the opcode. By describing the operational meanings of a program, an opcode enables us to predict the maliciousness of disassembled files in many cases. Previous work on word2vec-based long short-term memory (LSTM) [20] used opcode sequences and adopted an LSTM to consider the sequential order of the opcodes appearing in the file. They encoded opcodes using the word2vec embedding method

and then fed the result representation to the LSTM. Qiao et al. proposed converting grayscale images using word vector similarity calculated from opcodes and conveyed to a CNN for malware classification [21]. Zhang et al. used five machine-learning methods to classify ransomware families and applied statistical measures to opcode sequences to obtain latent information. They extracted n-gram sequences from an opcode sequence and selected meaningful n-gram sequences [8]. Then they calculated the TF-IDF [8] for the selected sequence and used n-gram sequences with high TF-IDF values as features for classification. Rather than the entire opcode sequence, some previous studies were based only on the API call sequences from the disassembled binary code. However, as mentioned earlier, malware developers adopt anti-disassembly techniques (e.g., obfuscation and packing techniques) to prevent malware from being analyzed. Therefore, always extracting high-level static features such as opcodes and API call sequences from malware samples is implausible.

### B. MALWARE CLASSIFICATION USING STATIC FEATURES WITHOUT DISASSEMBLY

Some existing studies on malware classification have been based on features extracted without disassembly. Among others, visualized malware images are commonly used as features for malware analysis and can be obtained directly from a binary file. These images are usually fed to a CNN [13], [14], [22] because CNNs exhibit outstanding performance in image classification tasks. In [13] and [22], CNN-based malware classification models were proposed to use a grayscale image generated by converting byte values between 0-255 into pixels as inputs. Xiao et al. [14] proposed a different visualization method that injects the structural information of malware in the PE format into malware images. Thus, they attempted to reduce malware family misclassification by distinguishing the different meanings of the same or similar pixel patterns from different structural layouts. They also adopted a unique two-phase approach that performs an SVM-based classification with the features provided by the CNN-based extractor. However, malware images suffer from an inherent limitation of information loss owing to image resizing in classification models. The distortion caused by obfuscated or encrypted features can twist distinct patterns that would otherwise be correctly recognized. Structural entropy is another promising feature that can be extracted from malware without disassembly. It is an entropy stream in which all the entropy values of the chunks of a binary file are calculated. Gibert et al. used the entropy stream transformed using a wavelet, and the entropy stream was then fed to the CNN-based malware classification model [15]. They empirically demonstrated that entropy streams are resilient to code obfuscation, although they lose detailed malware patterns. For instance, encryption causes some parts of malware to have high entropy, whereas the intentionally inserted dead code for obfuscation usually has extremely low entropy. To devise a robust method for

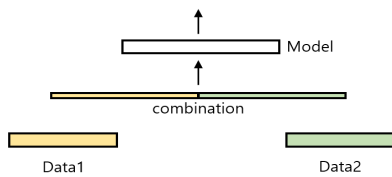


FIGURE 1. Early fusion.

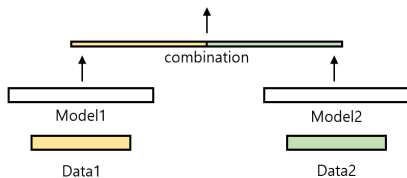


FIGURE 2. Late fusion.

malware family classification with no need for disassembling overhead, we combine malware images and structural entropy complementarily, which helps analyze modern malware armed with complicated anti-disassembly techniques.

### C. MALWARE CLASSIFICATION USING DYNAMIC FEATURES

Dynamic features represent the runtime behavior of a program. For instance, API calls, network traffic, and snapshots of registries are all dynamic features because they are gathered at runtime. Among these, API calls and opcode sequences are the most popular because malicious activities are eventually triggered by APIs and opcodes. Xue et al. [23] comprised two classifiers based on CNNs and other machine-learning techniques. Whereas the CNN-based classifier uses malware images, the other classifier uses variable n-grams of dynamic API call sequences earned at runtime. Malware language models [24], [25] were devised to predict the next system call based on the previous system calls. They treated API calls similar to normal language models treating words. Pektaş et al. used API call sequences for malware classification. They selected and used meaningful subsequences of API calls at runtime using a data mining technique instead of an entire sequence per malware sample [26]. Hansen et al. proposed a random forest malware classifier using statistical information about API calls as the input [27]. Pascanu et al. [24] proposed a recurrent neural network-based method using sequences of API calls for malware family classification. Unlike static features, the extraction of dynamic features requires malware samples to be executed in a virtualized environment (i.e., a sandbox). However, sophisticated malware has evasion functionality that causes the malware's execution to stop or behave differently when a virtualized environment is detected, making it difficult to obtain dynamic features from malware.

### D. MALWARE CLASSIFICATION BASED ON MULTI-MODAL LEARNING

Multi-modal learning [28] is a deep learning technique that learns multiple modalities simultaneously. The aim is to

maximize the amount of information by combining various modalities. The information that cannot be represented by one modality is complemented by other modalities [10]. Multi-modal learning combines various modalities in several ways. Early fusion [28] combined data from different modalities at a low level, as shown in Fig. 1. The fused data is fed to a classifier. In late fusion [28], the outputs of the individual classifiers on different modalities are combined, as shown in Fig. 2. Many models that use images and text simultaneously typically adopt multi-modal learning [29]. There are malware classification models based on multi-modal learning [4], [10], [30]. Gibert et al. [10] combined API calls, raw bytes, and opcodes using the late fusion method to exploit low-level and high-level information to enhance classification performance. Han et al. [4] combined static and dynamic information using an early fusion method. They used static API sequences extracted from disassembled files and dynamic API sequences logged during the runtime. In [30], the malware was identified by a multi-modal clustering algorithm using a PE header, string hashes, IAT, and byte entropy. Recently, cross-modal learning [29], [31], [32], a new multi-modal learning method that focuses on the adaptive combination of multiple modalities, has emerged. In the fields of computer vision and natural language processing, cross-modal learning has been studied to effectively combine multiple modalities [29], [31], [32]. In our classification model, a cross-modal attention mechanism was integrated as a data fusion method. This allows the data from the two multiple modalities of malware images and structural entropy to be flexibly combined to complement each other.

## III. OVERVIEW OF ATTENTION-BASED CROSS-MODAL CNN FOR MALWARE CLASSIFICATION

Our malware classification model is based on CNNs that learn and recognize the patterns of input data. CNNs are commonly used for malware detection and classification [16] because they effectively extract high-level features from low-level data without auxiliary feature engineering steps. As mentioned earlier, the proposed model uses malware images and structural entropy as inputs, which do not require disassembling malware. The proposed model combines input multi-modalities by calculating the cross-modal attention described later in this paper.

Fig. 3 shows the architecture of the proposed model. A malware image is considered to be a sequence of chunks of fixed byte size, similar to how structural entropy is represented as a sequence of entropy values of chunks. This allows cross-modal attention to combining a malware image and structural entropy by forcing each meaningful chunk to align independently with the sequences of different modalities. By calculating the cross-modal attention, our model can obtain reinforced features by extending the dimension of the features using information from another modality. A one-dimensional CNN (1D-CNN) was adopted to treat such a sequence of chunks. The 1D-CNN effectively processes sequences because the kernels of the 1D-CNN move



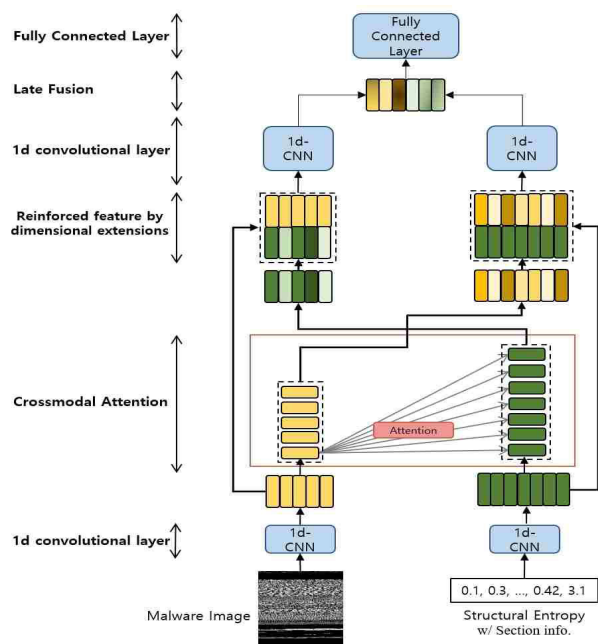


FIGURE 3. Architecture of attention-based cross-modal CNN.

only over elements of one input dimension. Each shallow 1D-CNN uses the corresponding modality of the input data. As shown in Fig. 3, the outputs of these shallow 1D CNNs were subsequently combined using the cross-modal attention mechanism.

Malware images and structural entropy are commonly used in malware classification; however, they do not yield high accuracy because of the aforementioned drawbacks. As shown in Fig. 4, our model reinforces each modality by combining malware images with structural entropy to complement each other.

The structural entropy is aligned at each element of the representation of the malware image (the yellow vector on the left side of Fig. 4) to overcome the disadvantages of malware images. Distortion by obfuscation of malware images is complemented by a structural entropy stream robust against obfuscation. In contrast, malware image information is aligned at each element of the representation of structural entropy (the green vector on the right side of Fig. 4) to overcome the disadvantage of structural entropy. In other words, the loss of byte chunks in structural entropy is mitigated using malware images. As a result, the dimensions of the latent representations of each modality were extended. The chunks of a feature in each modality are merged with the information in other modalities to implement the reinforced feature. Using these reinforced features, our model learns the pattern of multi-modal input data while complementing these two modalities.

However, combining these two modalities is difficult. One modality’s representation typically has a different length than that of another. This difference complicates the alignment of these two representations. Malware images and structural

entropy are both affected by their different lengths. This is because a malware image was downsampled to create an acceptable input format for our model. The proposed model effectively aligns the two representations of a malware image and structural entropy by applying an attention mechanism called cross-modal attention to each in this study. As shown in Fig. 4, cross-modal attention causes the two representations of different modalities to have the same length for each modality. In addition, the data for each modality were reinforced with the other modality data. For a malware image, the representation of the structural entropy shrinks to the same length as the malware image according to the degree of relevance of the individual chunks of the malware image. Similarly, the representation of a malware image is resized to that of structural entropy. Cross-modal attention obtains the relevance of both modalities and propagates it to the opposite modalities. As a result, the reinforced representations effectively represent the information of the chunks appearing simultaneously in both modalities.

The reinforced representations were then fed to their corresponding the following 1D-CNNs. The 1D-CNNs learn the patterns of each modality, along with the relevance of one modality to another. The outputs of the two 1D CNNs were concatenated into the final feature vector and passed through the fully connected layer.

The proposed method includes cross-modal attention-based early fusion that implements low-level interactions between “malware image” and “structural entropy” and late fusion that extracts and combines high-level information of each reinforced representation based on a CNN. An elaborately designed model can provide both early and late fusion advantages, effectively combining both modalities to identify malware variants of the same family with high performance.

In Sections IV, V, and VI, we describe the model components. In Section IV, we describe the malware image and structural entropy obtained using the preprocessing method. Section V describes the cross-modal attention process. Subsequently, we describe other components of the model.

#### IV. MULTI-MODAL INPUT DATA

##### A. MALWARE IMAGE

The proposed model learns two modalities using CNNs. The first modality is a malware image. A malware image is created by converting each byte of a binary file into a pixel between 0 and 255 without disassembling the file. Then, the image was downsampled to an appropriate input size for CNN. Fig. 5. shows an example of a malware image used in this study. Malware images can effectively represent the overall information of malware but have the disadvantage of being heavily influenced by obfuscation and encryption, which can cause image noise [13].

Most studies have used two-dimensional CNNs (2D CNNs) for malware image analysis [13], in which the kernels move in two-dimensional directions. In contrast, we use a 1D-CNN, in which the kernels move in one direction. This

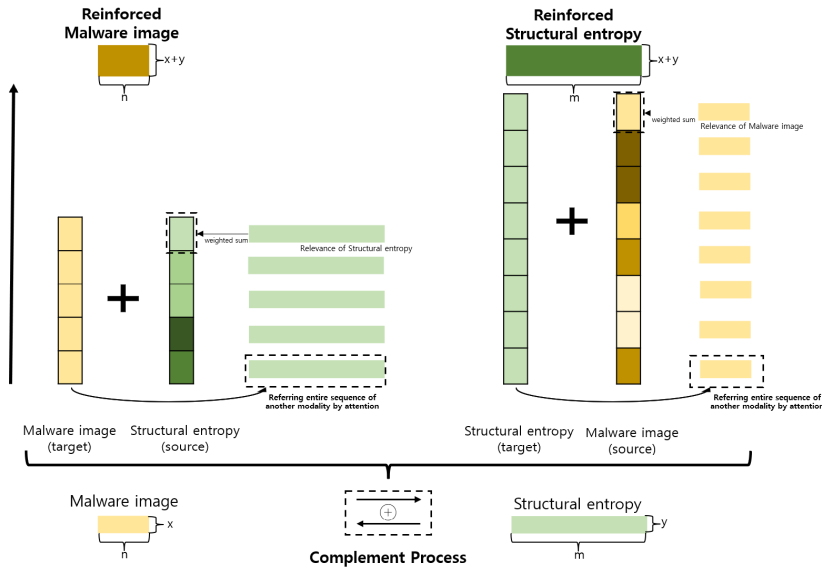


FIGURE 4. Reinforcement of modality by modality combination.

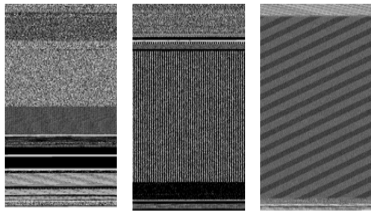


FIGURE 5. Malware image examples.

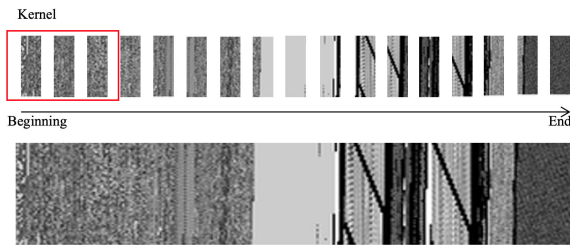


FIGURE 6. How to move the kernel in the byte-sequence direction(1D-CNN).

allows these features to be combined with the results of structural entropy feature engineering in a direction along the image, despite their different modalities. However, the two-dimensional movement directions of kernels in a 2D-CNN complicate the alignment between the two features of different modalities. In addition, our internal experiment shows that the malware family classification model with a 1D convolutional layer yields an accuracy as high as that of the existing 2D-CNN-based classification model.

In summary, we created a malware image from a raw malware binary file by converting each byte into a pixel value between 0-255. The image was then downsampled into a

resized file of 64C-784 pixels. The kernels in the 1D-CNN directly access the image in one direction to create low-level features.

**B. STRUCTURAL ENTROPY WITH SECTION INFORMATION**

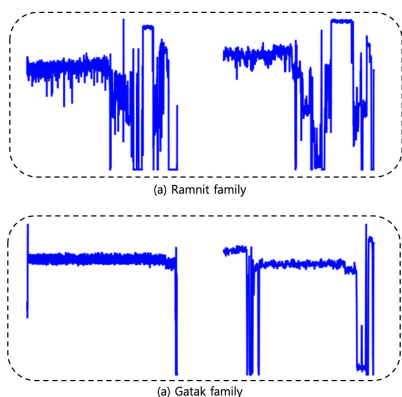
The structural entropy embedding section information is the second modality we proposed.

Typically, structural entropy is a sequence of entropy values in the chunks of a file. Structural entropy is calculated based on the frequency of bytes and has the advantage of being robust to obfuscation [15]. To obtain the structural entropy, we divided the malware into chunks and calculated and extracted the Shannon entropy [33] based on the frequency of bytes in each chunk. The calculation formula is as follows.

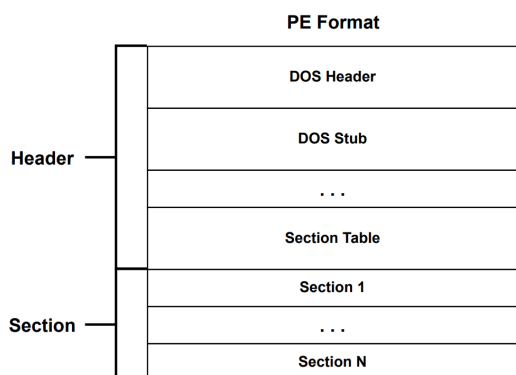
$$H(x) = -\sum_{j=1}^n p(x_j) \log_2 p(x_j) \tag{1}$$

In (1), n is the maximum value of the byte and is determined to be 255. X is the byte value from 0 to 255, and  $p(x_j)$  is the probability of occurrence within the chunk of X. The patterns of structural entropy [14] in the same malware family tend to be similar. Fig. 7 depicts the four structural entropies of malware samples from the Ramnit and Gatak families. It was found that malware samples from the same family had similar shapes.

An executable file in PE format is structured in the form of headers and sections for execution, as shown in Fig. 8. Sections serve specific functions during execution. Normally, each section has a well-known section name, such as .text or .data. Malware samples belonging to the same family could have similar section information [14]. In addition, malware in PE format is usually accompanied by a normal section and various other sections with unfamiliar names. Thus, section



**FIGURE 7.** Structural entropy samples of the same family show similar patterns: (a) two Ramnit samples have a series of the relatively stable values around the head while irregular values around the tail. (b) Both the samples from Gatak family have clear horizontal horizontal lines in the middle.

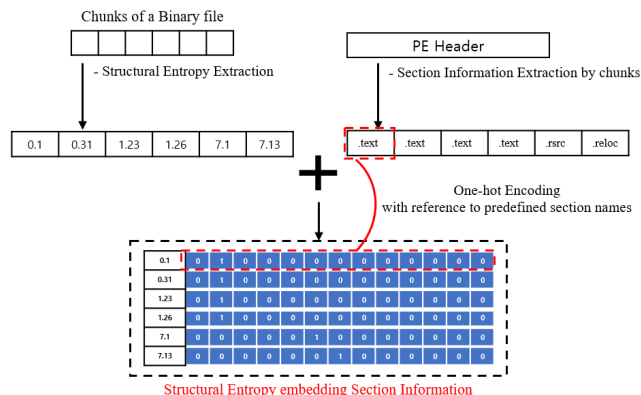


**FIGURE 8.** PE format of exe file: the section table of the file header is an array of section headers. A section name is defined in the corresponding section header.

information can help discriminate malware from normal executable files.

Unfortunately, typical structural entropy does not contain section information in malware. To overcome this limitation, we append the corresponding section information to the structural entropy as one-hot encoding, as suggested in [34] (shown in Fig. 9.) We categorized section names into “standard (or normal) section names” and “undefined sections.” For standard sections, such as Header, .text, .data, .idata, .rdata, .rsrc, .reloc, .bss, .edata, .tls, .xdata, .pdata, .sdata, we adopted section names as section information. For the other sections that mostly appear in malware, detailed section names are omitted from the section information, but we encode only the information that it is an “undefined section.” This is because the detailed names of the undefined section might be meaningless because the section names in malware do not follow naming conventions and are intentionally distorted in many cases.

In summary, a chunk is represented as an entropy value and a one-hot vector for the 14 sections. Note that the structural entropy combined with the section information should be padded or truncated to a fixed length before being fed to



**FIGURE 9.** Feature engineering method [34].

our CNN model. Consequently, the structural entropy of the input is represented as a vector of  $15 \times length$ . The  $length$  is the fixed maximum length for feeding the CNNs structural entropies.

### V. CROSS-MODAL ATTENTION

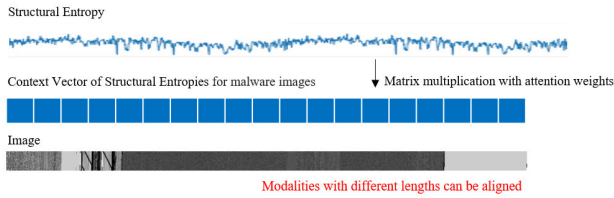
This section describes the attention mechanism and the architecture of the proposed cross-modal attention in detail. The 1D-CNN layers for malware images and structural entropies described in the previous section are followed by the process of combining the two types of features. Malware images and structural entropy can compensate for each other when used simultaneously. Additionally, structural entropy is more robust against code obfuscation attacks than malware images, whereas malicious code patterns are usually found in binary code images rather than structural entropies.

We adopted an attention mechanism for effectively combining these two modalities [35]. It is a concept proposed in the field of machine translation from a source sentence (i.e., input sequence) to a target sentence (i.e., output sequence). It flexibly calculates the relevance between the input and output sequences. The relevance score of the source data  $h$  and target data  $S$  is determined by the attention vector  $W_a$  using the score function below. The weights of  $W_a$  were learned during the training phase.

$$score(S, h) = Softmax(h \times W_a \times S^T) \quad (2)$$

Finally, the context vector [35] for the target data  $S$  is obtained by multiplying the relevance score by the source data  $h$ , which means that the attention to  $h$  for  $S$  is calculated. The attention indicates the relevance of the entire sequence of source data based on the inner product operation.

In our model, cross-modal attention aligns the information on a malware image for structural entropy windows and structural entropy for malware image windows. As shown in Fig. 10, sequences of different lengths were aligned when calculating the attention. In the figure, the feature sequence extracted from a malware image is shorter than the feature sequence in terms of structural entropy. Because the cross-modal attention mechanism calculates the context



**FIGURE 10.** Example of relevance score calculation of structural entropy(source) to malware image(target) for alignment.

vectors of each element of the structural entropy (for the image), the context vectors' length is defined to match the malware image length. This indicates that the use of cross-modal attention is independent of the lengths of the two source and target sequences.

Fig. 11 shows a portion of the process in which the imaging modality is reinforced using the relevant information of the structural entropy from the attention mechanism. The features of the solid box in the center depict how to address the structural entropy for the individual windows of a malware image. The red matrix in the center is the attention weight used to obtain the relevance score of the structural entropy for the malware image. The numbers in red correspond to those in the equations below. We assume that malware image sequence  $x$  and structural entropy  $y$  are combined with cross-modal attention. The sequences  $x$  and  $y$  are represented as follows.

$$x = [x^1, x^2, \dots, x^{784}] \in \mathbb{R}^{784 \times 64} \quad (3)$$

$$y = [y^1, y^2, \dots, y^{length}] \in \mathbb{R}^{length \times 15} \quad (4)$$

Each  $x$  and  $y$  pass through the 1D convolution layers, where 64 kernels of size one convolve. Local features in the individual windows belonging to a malware image or structural entropy are extracted and embedded using 1D convolution layers. Each element in the resulting sequences has the same size of 64, forming two sequences of size  $784 \times 64$  for a malware image and of  $length \times 64$  for a structural entropy, as shown in (5) and (6).

$$X = Conv1D(x, k_{64}) \in \mathbb{R}^{784 \times 64} \quad (5)$$

$$Y = Conv1D(y, k_{64}) \in \mathbb{R}^{length \times 64} \quad (6)$$

Next, sequence  $X$  attends to sequence  $Y$  using the cross-modal attention mechanism, and vice versa. We explain the cross-modal attention in the case of one attention direction; the target sequence is  $X$ , and the source sequence is  $Y$ . Equation (7) indicates the calculation of the relevance of sequence  $Y$  to sequence  $X$ . First, we obtain the scores with the score function, in which  $W_a$  denotes the alignment between sequences  $X$  and  $Y$ . Then, the relevance score  $A$  is determined by normalizing the result of the score function with a softmax function.

$$\begin{aligned} A &= score(X, Y) \\ W_a &\in \mathbb{R}^{64 \times 64} \\ score(X, Y) &= Softmax(Y \times W_a \times X^T) \end{aligned} \quad (7)$$

The next step is to obtain a context vector, a weighted vector of the structural entropy of the malware image. The context vector is calculated using Equation (8). The relevant information of  $Y$  for  $X$  is summarized by multiplying the attention weight and  $Y$ . Consequently, the context vector is a weighted sum of the sequence data  $Y$  aligned for a window of  $X$ .

$$C = Y^T \times A \quad (8)$$

Finally, the context vectors were concatenated to the corresponding windows of  $X$ , as shown in (9). Each of the final vectors contains the information of a window of  $X$  and the information of  $Y$  relevant to the window of  $X$ .

$$R_X = [X, C] \quad (9)$$

In the above operation, the relevant information from  $Y$  to  $X$  is extracted as a context vector using the cross-modal attention mechanism and stacked on  $X$ . In other words, the malware image modality is reinforced by the attention mechanism. Hereby, the following 1D-CNNs can generate higher-level information from raw byte sequences of malware images and entropy value sequences of structural entropy. This attention operation was also performed in the opposite direction. The structural entropy modality learns the structural entropy and malware images aligned in the window step of structural entropy simultaneously. Therefore, cross-modal attention effectively combines the two modalities with each other in a cross manner.

## VI. 1D-CONV FOR HIGH-LEVEL FEATURE EXTRACTION AND LATE FUSION

Modality reinforced with cross-modal attention is represented as a 128-dimensional vector. The reinforced features for the two modalities were fed to the 1D CNNs to extract the higher-level features. Fig. 12 shows the architecture of the 1D-CNN for both modalities.

The 1D CNN had four stacked 1D convolution layers, each of which used 70 filters of size 3. The output of each convolution layer was passed through a batch normalization layer, followed by the ReLU activation function. Finally, the max-pooling layer with a pooling size of  $2 \times 1$  and a stride of 2 is the most outstanding feature among the features activated by the ReLU activation function.

As shown on the right side of Fig. 11, the features that passed through the 1D convolution layers are concatenated. The concatenated vector is then fed to the following two fully connected layers, each comprising 1000 and 300 nodes. The last layer is structured to classify malware families using a softmax layer.

## VII. TRAINING AND TESTING PHASE

The proposed attention-based cross-modal CNN is an end-to-end deep learning framework. It has the advantage of not requiring separate training steps for feature extraction in different modalities.



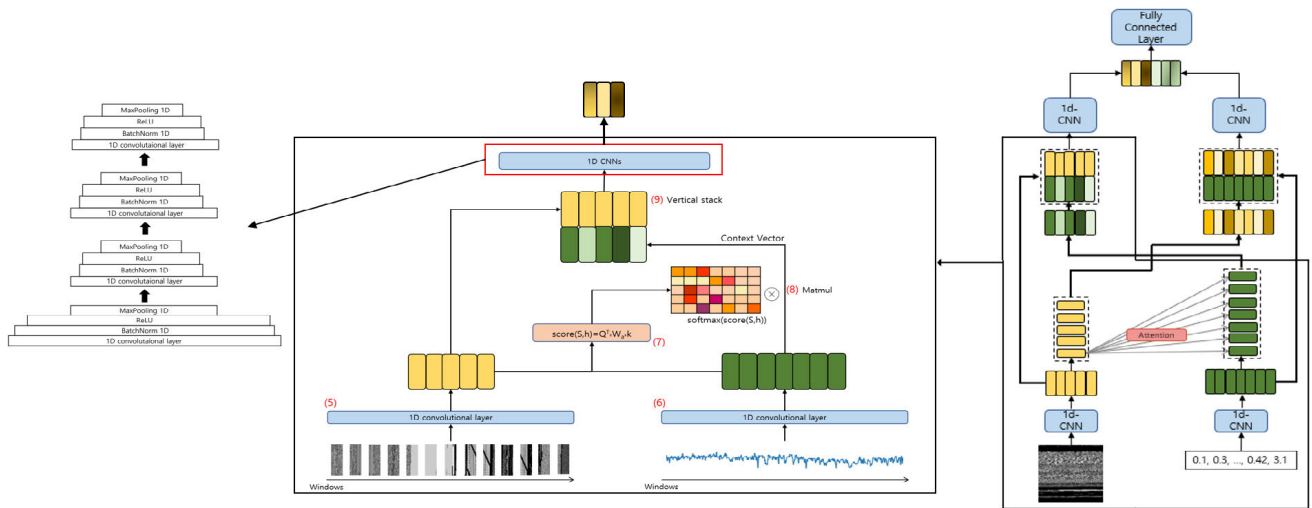


FIGURE 11. Architecture of the proposed model in detail and process of attention to structural entropy (the red numbers in the figure denote the corresponding equation numbers in Section V).

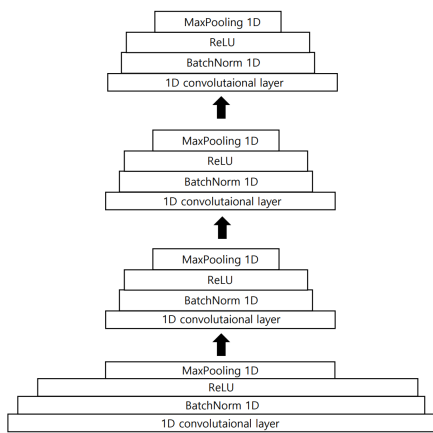


FIGURE 12. Architecture of 1D CNN for higher-level feature extraction.

The entire process of the proposed model comprises training and testing phases. In the training phase, multiple data points were computed using cross-modal attention to complement each other. Our model classifies malware into families using a CNN that extracts high-level features and fully connected layers, as described in Sections IV, V, and VI. Consequently, the weights in our model were updated. Our model was trained based on the selected hyperparameters listed in Table 1.

Next, we evaluate the performance of our model during the test phase. In the test phase, our model, whose weights were trained in the training phase, classified malware into families based on the dataset. The k-fold validation method was adopted for evaluation. K-fold validation is a cross-validation method used when the number of samples is small. It involves dividing the dataset into k subsets and then using k-1 subsets as training datasets and one subset as the test dataset. A 10-fold validation method was adopted for the experiment.

TABLE 1. Hyperparameters of proposed models.

Parameter	Cross-modal CNN
Batch Size	128
Epochs	80
Learning Rate	1e-4 to 1e-5
Loss Function	Cross entropy
Optimizer	Adam

TABLE 2. Detail of Microsoft Malware Classification Challenge dataset.

Family	ClassID	#numbers
Ramnit	0	1541
Lollipop	1	2478
Kelihos_ver3	2	2942
Vundo	3	475
Simda	4	42
Tracur	5	751
Kelihos_ver1	6	398
Obfuscator.ACY	7	1228
Gatak	8	1013

## VIII. EXPERIMENTS

In this section, we describe the dataset and evaluate the performance of our proposed model. As mentioned, our model is evaluated on the Kaggle Microsoft Malware Classification Dataset [17], Malimg [18], and BODMAS [19] datasets. We evaluated our model by implementing a baseline model, especially for the Kaggle Microsoft Malware Classification Dataset, and compared the performance results with those of other studies using this dataset. The details of this are described in the following sections. For the experiment, we implemented the proposed model using the PyTorch deep learning framework. The GPU used for training was a GeForce RTX 3090 with 24 GB of memory.

TABLE 3. Detail of Maling dataset.

Family	ClassID	#numbers
Allaple.L	0	22
Allaple.A	1	116
Yuner.A	2	2816
Lolyda.AA 1	3	1568
Lolyda.AA 2	4	193
Lolyda.AA 3	5	104
C2Lop.P	6	144
C2Lop.gen!g	7	166
Instantaccess	8	177
Swizzot.gen!I	9	162
Swizzor.gen!E	10	320
VB.AT	11	52
Fakerean	12	213
Alueron.gen!J	13	183
Malex.gen!J	14	122
Lolyda.AT	15	156
Adialer.C	16	17
Wintrim.BX	17	16
Dialplatform.B	18	148
Dontovo.A	19	80
Obfuscator.AD	20	126
Agent.FYI	21	132
Autorun.K	22	122
Rbot!gen	23	94
Skintrim.N	24	797

The performance metrics are accuracy and the F1-score, a harmonic mean of precision and recall. Each performance metric was calculated on a confusion matrix comprising true positive (TP), true negative (TN), false positive (FP), and false negative (FN) values. Accuracy is calculated as  $TP+TN/(TP+FP+TN+FN)$ , precision as  $TP/(TP+FP)$ , recall as  $TP/(TP+FN)$ , and F1-score as  $2 \cdot C- (recall \cdot precision)/(recall+precision)$ .

## A. DATASET

### 1) MICROSOFT MALWARE CLASSIFICATION CHALLENGE DATASET

We used the Kaggle Microsoft Malware Classification Challenge (BIG-2015) dataset [17] to train and evaluate the performance of our model. Microsoft Malware Classification Challenge dataset [17] contains 10,868 malware assemblies and binary files from nine families. Table 2 lists the number of samples in the Microsoft Malware Classification Challenge dataset. The bytes of the binary files in Microsoft Malware Classification Challenge dataset are converted into pixels and downsampled to generate  $64 \times 784$  malware images. At this time, “??” bytes are the bytes that cannot be identified, and in this experiment, we replaced them with FF. Structural entropy was calculated from the byte values in the binary file. Our model uses structural entropy embedding section

TABLE 4. Detail of BODMAS-14 dataset.

Family	ClassID	#numbers
sfone	0	4729
wacatac	1	4694
upatre	2	3901
wabot	3	3673
small	4	3339
ganelp	5	2232
dinwod	6	2057
mira	7	1960
berbew	8	1749
sillyp2p	9	1616
ceeinject	10	1169
gepys	11	1124
benjamin	12	1071
musecador	13	1054

information, which can be extracted from the file header. Unfortunately, all header information is missing in Microsoft Malware Classification Challenge dataset. Accordingly, the structural entropy of the input is represented as a vector of  $14 \times length$ . To deal with this limitation, we use the assembly file that BIG 2015 provides, ending up appending section information. The maximum length is fixed at 3600 for feeds into CNN.

### 2) MALIMG DATASET

The Maling Dataset [18] contains 9,339 malware samples from 25 families. These were provided in the form of visualized images. To exploit these two malware modalities, we need malware images and their structural entropy. However, we could not collect approximately 1000 malware binaries from the Maling Dataset from VirusTotal. Therefore, 8,250 binaries were used in this experiment. Table 3 lists the number of samples in the collected binaries. Malware images and structural entropies with information about the respective sections were created from the malware binaries. The maximum length was fixed at 873 for feeds into CNN.

### 3) BODMAS-14 DATASET

We use the BODMAS [19] dataset to evaluate the model's performance. BODMAS [19] dataset is the most recent open malware dataset. A total of 57,293 malware binaries and information on 581 malware families were included. In the case of some families, the number of samples was small. Therefore, in this study, 14 families with more than 1000 samples were used, named BODMAS-14 in this experiment. The malware images and structural entropies were extracted from the byte value of the file, and section information was extracted based on the header information and one-hot encoded to generate the structural entropy. The header information was not accessible to binary files. In this case, all sections were treated as unknown. The maximum length was fixed at 4000 to be fed into the CNN. In total, 34,368 binaries were used in the experiment. Table 4 lists the details of the samples used in the experiments.

**TABLE 5. Malware classification results of baseline models on the microsoft Malware classification challenge dataset.**

Model	Used Modality	Acc	F1-score
B1	Structural Entropy (w/ section info)	97.95	0.936
B2	Malware Image	96.79	0.926
B3	Structural Entropy and Malware Image	98.28	0.960
Proposed	Structural Entropy and Malware Image (Using cross-modal attention)	<b>98.72</b>	<b>0.980</b>

**TABLE 6. Malware classification results in comparison with other research on the microsoft Malware classification challenge dataset.**

Author	Method and used Modality	Acc	F1-score	Does it require disassembly step?
Ours	<b>Attention-based Cross-modal CNN</b> (Proposed model)	98.72	0.9800	No
M.Xiao (2021)	VGG16 [14] (CoLab)	98.94	N/A	No
Gibert <i>et al.</i> (2020)	Multi-modal Network [10] (API, Opcode, Byte)	99.75	0.9954	Yes
Jiaqi Yan <i>et al.</i> (2019)	CNN [36] (Control flow graph)	99.25	N/A	Yes
Gibert <i>et al.</i> (2018)	CNN [13] (128 × 128 Grayscale Image)	97.50	0.9400	No
Gibert <i>et al.</i> (2018)	CNN [15] (Structural Entropy with wavelet transform)	98.28	0.9636	No
Mitchell Mays (2017)	CNN and NN [37] (Opcode n-gram, image)	97.70	N/A	Yes
Y. Zhang <i>et al.</i> (2016)	Ensemble Network [38] (Opcode, API, Section)	99.74	0.9938	Yes
Mansour Ahmadi (2016)	XGBoost [39] (API feature vector)	98.43	N/A	No

## B. RESULTS ON MICROSOFT MALWARE CLASSIFICATION CHALLENGE DATASET

### 1) PERFORMANCE COMPARISON OF BASELINE MODELS

For evaluation, we compared our model with other malware classification models that use malware images or structural entropy. We implemented malware classification models as a baseline by benchmarking the architectures proposed in other studies [13], [15]. We implemented a late fusion model without cross-modal attention that combines malware images and structural entropy with late fusion to prove the need for cross-modal attention.

Table 5 shows a performance comparison between our attention-based cross-modal CNN and the baseline models. Baselines 1 (B1) and 2 (B2) are the CNN models trained using structural entropy and malware images, respectively. Model B1 was trained using structural entropy, which appends the section information of the file using the method proposed in [34]. Model B3 is a multi-modal CNN model with a late fusion method in which cross-modal attention is removed from our model.

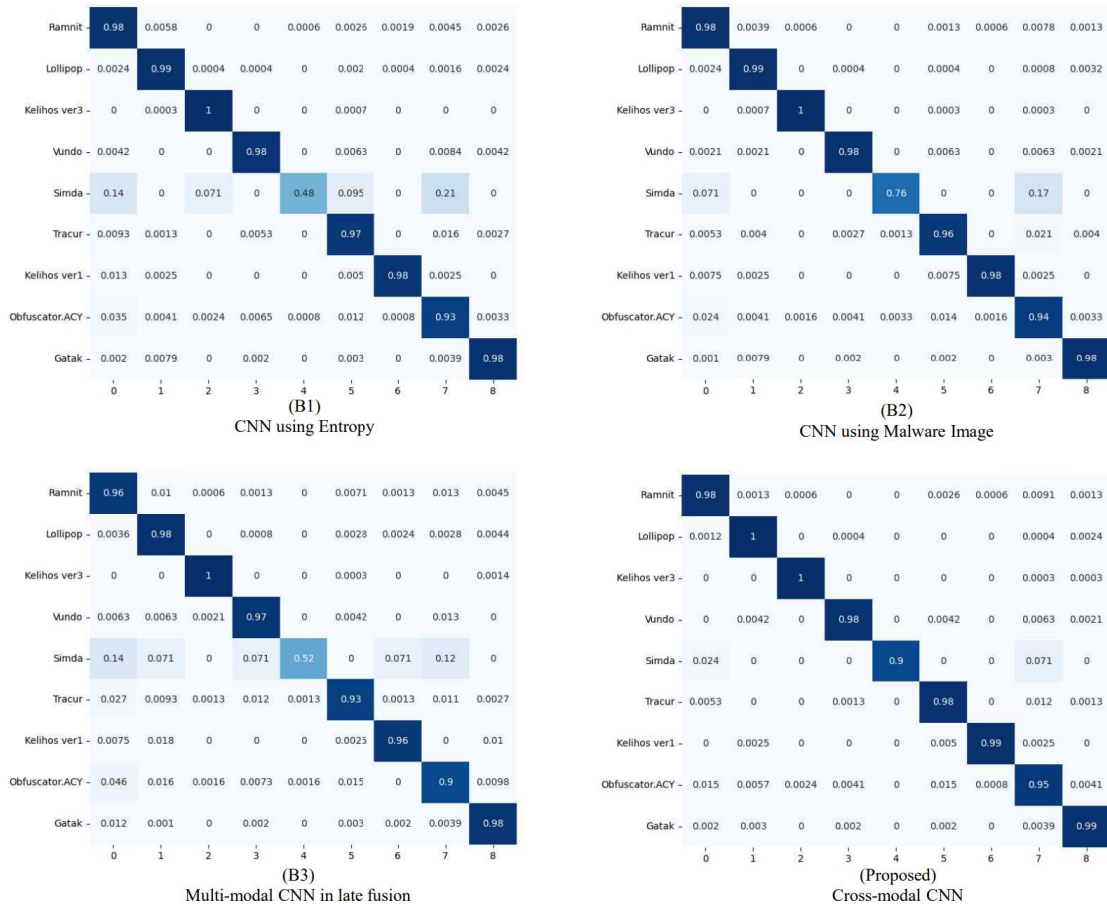
As shown in Table 5, the proposed model classifies malware into families with high performance. The accuracy was improved up to 1.93, and the F1-score was improved up to 0.054 compared to the models trained only with one modality (malware image or structural entropy). The proposed attention-based cross-modal CNN classifies malware

with a higher performance than model B3. These results prove that the proposed cross-modal attention is an effective fusion method for compensating for information from malware images and structural entropy.

Fig. 13. shows the confusion matrix generated based on the prediction results of each baseline model. Fig. 13. shows that our cross-modal CNN performed significantly better than the other models. For the Simda class with a small number of samples, B1 and B2 classified instances with a low accuracy of approximately 0.5, whereas the proposed cross-modal CNN classifies them with an accuracy of 0.904. B3 also performed better for the Simda class than when only one modality was used but showed lower accuracy than our cross-modal CNN concerning the Simda class. In addition, our model classifies the obfuscator. ACY class with the highest accuracy.

### 2) PERFORMANCE COMPARISON WITH OTHER RESEARCH

Several studies have proposed methods for classifying malware families. In this section, we compare the malware classification models of previous studies, with and without disassembly, to our attention-based cross-modal CNN. Table 6 shows the performance comparison between the models, and all the compared models were evaluated with 10-fold validation on Microsoft Malware Classification Challenge dataset.



**FIGURE 13.** Confusion Matrix of baseline models on Microsoft Malware Classification Challenge dataset (A matrix with a darker main diagonal is a better confusion matrix).

**TABLE 7.** Malware classification results of baseline models on the Maling dataset.

Model	Used Modality	Acc	F1-score
B4	Structural Entropy (w/ section info)	97.15	0.914
B5	Malware Image	98.63	0.947
B6	Structural Entropy and Malware Image	98.86	0.971
Proposed	Structural Entropy and Malware Image (Using cross-modal attention)	<b>99.09</b>	<b>0.976</b>

Gibert et al. used command frequency, API, and types of sections appearing in a file for malware classification [10]. Mays et al. [37] classified malware families based on n-gram instructions and malware-image data. In addition, the Hydra [10] multi-modal model classified malware into families using APIs, instruction sequences, and byte sequences. As listed in Table 6, the aforementioned studies used static features with disassembly.

In [13] and [14], malware families were classified using malware images. In particular, CoLab, which appends section information to binary files lacking malware images, was

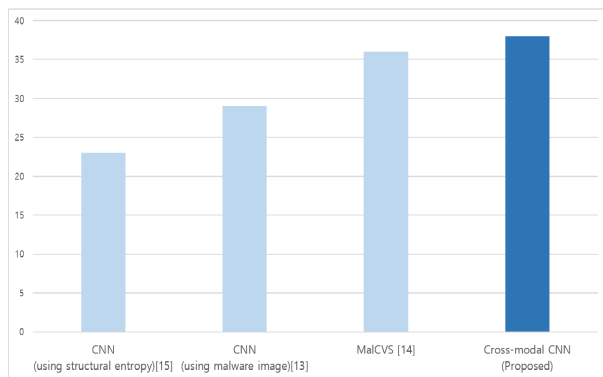
proposed in [14]. MalCVS, trained using CoLab, classified malware with higher performance when using normal malware images. Gibert et al. [15] classified malware families using structural entropy, which was applied using a wavelet transform. The XGBoost in [39] classifies malware families based on the API of binary files. These studies classified malware samples into families despite not using disassembly.

As shown in Table 6, the use of the extracted features from disassembled malware enables the classification models to exhibit higher performance than the models based on non-disassembled malware because disassembled codes have



**TABLE 8. Malware classification results in comparison with other research on the Maling dataset.**

Author	Method (The number of samples in dataset)	Acc	F1-score	Does it require disassembly
Ours	<b>Attention-based Cross-modal CNN (8,250)</b>	99.09	0.976	No
Rikima Mitsuhashi (2022)	CNN [40] (9,339)	98.96	N/A	No
Jin Ho Go <i>et al.</i> (2020)	CNN [41] (9,339)	98.86	N/A	No
YA-SHU LIU <i>et al.</i> (2019)	CNN [42] (9,458)	99.00	N/A	No
R. VINAYAKUMAR <i>et al.</i> (2018)	CNN [43] (9,339)	96.30	0.9620	No
L. Nataraj (2011)	Distance-Metric [18] (9,458)	97.18	N/A	No

**FIGURE 14. The number of true positives on Simda class.**

higher-level semantic information than non-disassembled codes. Nevertheless, our model with cross-modal attention achieved a competitive high accuracy of 98.72. Above all, the malware images and structural entropy used in our model were extracted without a disassembly step.

Most importantly, owing to the use of multi-modal features, our model was more robust against data imbalances. As shown in Fig. 14, for the Simda class, the number of true positives in our model was 38 out of 42 samples, demonstrating an accuracy of 90.4, which is the highest accuracy compared to models from other studies [13], [14], [15] based on unimodal feature.

### C. RESULTS ON MALING DATASET

#### 1) PERFORMANCE COMPARISON OF BASELINE MODELS

We implemented B4, B5, and B6 baseline models to compare the performance of our model as described in the previous section. Table 7 presents a performance comparison between the proposed and baseline models. The CNN models, baselines 4 (B4) and 5 (B5) were trained using structural entropy, which appended section information and malware images, respectively. Baseline 6 (B6) is a multi-modal CNN model with a late fusion method that uses structural entropy and malware images. Our proposed model showed the highest performance for all the performance metrics. The accuracy was 99.09, and the F1-score was 0.976. As confirmed in the previous section, we infer that feature fusion using cross-modal

attention is more effective at combining the two modalities than a simple concatenation of malware images and structural entropy when our model is compared with B6.

Because of the many malware families in the Maling dataset, we excluded a confusion matrix on Maling. What is noteworthy in this experiment is the outstanding achievement of our model for packed samples. Most samples in the Yuner class of the Maling dataset were packaged using UPX packers. Our model perfectly classified the packed samples in the Yuner A class using only binaries, resulting in an accuracy of 100%.

#### 2) PERFORMANCE COMPARISON WITH OTHER RESEARCH

In this section, we compare our model with other malware classification studies using Maling datasets. Table 8 shows the performance comparison between the models, and all the compared models were evaluated with 10-fold validation on the Maling Dataset.

Various malware classification studies using Maling have been conducted. Nataraj *et al.* [18] first proposed the Maling dataset. In a recent study [40], Mitsuhashi and Shinagawa proposed a malware classification model based on a pre-trained model [40].

Table 8 shows the accuracy and F1-score values of six studies. Unfortunately, our model cannot be compared directly to the others. We failed to collect all the binaries corresponding to the malware images in the Maling dataset. As such, we could not extract structural entropy from some of the Maling datasets. We specified in Table 8 the number of samples used in each study. Nevertheless, the proposed model is considered a reasonable method; it shows the highest accuracy and f1-score. This proves that structural entropy accelerates performance by compensating for information that malware images cannot express.

### D. RESULTS ON BODMAS-14 DATASET

#### 1) PERFORMANCE COMPARISON OF BASELINE MODELS

We implemented the B7, B8, and B9 baseline models to compare their performance, as described in the previous section. Table 9 presents a performance comparison between the proposed model and baseline models. The performance of the proposed model was the best. In addition, Table 9 shows

**TABLE 9. Malware classification results in comparison of baseline models on the BODMAS-14 dataset.**

Model	Used Modality	Acc	F1-score
B7	Structural Entropy (w/ section info)	87.40	0.8963
B8	Malware Image	87.94	0.8831
B9	Structural Entropy and Malware Image	92.57	0.9380
Proposed	Structural Entropy and Malware Image (Using cross-modal attention)	<b>92.92</b>	<b>0.9415</b>

higher performance in terms of accuracy and the f1-score than when only the malware image or structural entropy was used as training data. As a result, we obtained results similar to those in the previous sections for the latest malware dataset. However, no other study has used the BODMAS-14 dataset in a configuration comparable to ours, so a performance comparison with other studies could not be performed.

## IX. CONCLUSION

In this paper, we propose a cross-modal CNN that classifies malware families with high performance. The proposed model is a multi-modal learning-based network simultaneously trained using malware images and structural entropy. Using cross-modal attention, two modalities with different granularities are reinforced to compensate for each other's drawbacks. Our model achieved high accuracy for malware families with three different datasets, even when non-disassembled malware was used. The experimental results demonstrate that multiple modalities perform better than one modality and that our cross-modal attention for feature fusion is more effective than simple fusion. The code of our model is available at "<https://github.com/PLASLaboratory/Attention-based-crossmodal-CNN-using-Non-disassembled-Feature-for-Malware-Classification>".

Our future work will involve the addition of more modalities. ASCII strings, IP addresses, and import table are extracted from binary code without disassembly, and they have semantic information that malware image and structural entropy do not contain. We plan to make our model accommodate these features by considering a new modality.

## REFERENCES

- [1] (2021). *Picus Security*. [Online]. Available: <https://www.picussecurity.com/resource/blog/red-report-2021-top-ten-attack-techniques>
- [2] A. Shabtai, R. Moskovitch, Y. Elovici, and C. Glezer, "Detection of malicious code by applying machine learning classifiers on static features: A state-of-the-art survey," *Inf. Secur. Tech. Rep.*, vol. 14, no. 1, pp. 16–29, 2009. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1363412709000041>
- [3] A. Abusitta, M. Q. Li, and B. C. M. Fung, "Malware classification and composition analysis: A survey of recent developments," *J. Inf. Secur. Appl.*, vol. 59, Jun. 2021, Art. no. 102828. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2214212621000648>
- [4] W. Han, J. Xue, Y. Wang, L. Huang, Z. Kong, and L. Mao, "MalDAE: Detecting and explaining malware based on correlation and fusion of static and dynamic characteristics," *Comput. Secur.*, vol. 83, pp. 208–233, Jun. 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S016740481831246X>
- [5] M. Sikorski and A. Honig, *Practical Malware Analysis: The Hands-on Guide to Dissecting Malicious Software*. San Francisco, CA, USA: No Starch Press, 2012.
- [6] A. Afianian, S. Niksefat, B. Sadeghiyan, and D. Baptiste, "Malware dynamic analysis evasion techniques: A survey," *ACM Comput. Surveys*, vol. 52, no. 6, pp. 1–28, Nov. 2020.
- [7] M. Hassen, M. M. Carvalho, and P. K. Chan, "Malware classification using static analysis based features," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Nov. 2017, pp. 1–7.
- [8] H. Zhang, X. Xiao, F. Mercaldo, S. Ni, F. Martinelli, and A. K. Sangaiah, "Classification of ransomware families with machine learning based on N-gram of opcodes," *Future Gener. Comput. Syst.*, vol. 90, pp. 211–221, Jan. 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167739X18307325>
- [9] *Hex Ray, IDA Pro-Hex Rays*. Accessed: Mar. 7, 2023. [Online]. Available: <https://www.hex-rays.com/ida-pro/>
- [10] D. Gibert, C. Mateu, and J. Planes, "HYDRA: A multimodal deep learning framework for malware classification," *Comput. Secur.*, vol. 95, Aug. 2020, Art. no. 101873. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167404820301462>
- [11] D. Gibert, C. Mateu, and J. Planes, "Orthrus: A bimodal learning architecture for malware classification," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–8.
- [12] X. Chong, Y. Gao, R. Zhang, J. Liu, X. Huang, and J. Zhao, "Classification of malware families based on efficient-net and 1D-CNN fusion," *Electronics*, vol. 11, no. 19, p. 3064, Sep. 2022.
- [13] D. Gibert, C. Mateu, J. Planes, and R. Vicens, "Using convolutional neural networks for classification of malware represented as images," *J. Comput. Virol. Hacking Techn.*, vol. 15, no. 1, pp. 15–28, Mar. 2019.
- [14] M. Xiao, C. Guo, G. Shen, Y. Cui, and C. Jiang, "Image-based malware classification using section distribution information," *Comput. Secur.*, vol. 110, Nov. 2021, Art. no. 102420.
- [15] D. Gibert, C. Mateu, J. Planes, and R. Vicens, "Classification of malware by using structural entropy on convolutional neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 1–6.
- [16] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *Proc. Int. Conf. Eng. Technol. (ICET)*, Aug. 2017, pp. 1–6.
- [17] R. Ronen, M. Radu, C. Feuerstein, E. Yom-Tov, and M. Ahmadi, "Microsoft malware classification challenge," 2018, *arXiv:1802.10135*.
- [18] L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath, "Malware images: Visualization and automatic classification," in *Proc. 8th Int. Symp. Visualizat. Cyber Secur.* New York, NY, USA: Association for Computing Machinery, Jul. 2011, pp. 1–7, doi: [10.1145/2016904.2016908](https://doi.org/10.1145/2016904.2016908).
- [19] L. Yang, A. Ciptadi, I. Laziuk, A. Ahmadzadeh, and G. Wang, "BODMAS: An open dataset for learning based temporal analysis of PE malware," in *Proc. IEEE Secur. Privacy Workshops (SPW)*, May 2021, pp. 78–84.
- [20] J. Kang, S. Jang, S. Li, Y.-S. Jeong, and Y. Sung, "Long short-term memory-based malware classification method for information security," *Comput. Elect. Eng.*, vol. 77, pp. 366–375, Jul. 2019.
- [21] Y. Qiao, W. Zhang, X. Du, and M. Guizani, "Malware classification based on multilayer perception and Word2Vec for IoT security," *ACM Trans. Internet Technol.*, vol. 22, no. 1, pp. 1–22, Sep. 2021, doi: [10.1145/3436751](https://doi.org/10.1145/3436751).
- [22] A. Bensaoud, N. Abudawaood, and J. Kalita, "Classifying malware images with convolutional neural network models," *Int. J. Netw. Secur.*, vol. 22, no. 6, pp. 1022–1031, Oct. 2020.
- [23] D. Xue, J. Li, T. Lv, W. Wu, and J. Wang, "Malware classification using probability scoring and machine learning," *IEEE Access*, vol. 7, pp. 91641–91656, 2019.

[24] R. Pascanu, J. W. Stokes, H. Sanossian, M. Marinescu, and A. Thomas, "Malware classification with recurrent networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2015, pp. 1916–1920.

[25] B. Athiwaratkun and J. W. Stokes, "Malware classification with LSTM and GRU language models and a character-level CNN," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2017, pp. 2482–2486.

[26] A. Pektas and T. Acarman, "Malware classification based on API calls and behaviour analysis," *IET Inf. Secur.*, vol. 12, no. 2, pp. 107–117, Mar. 2018.

[27] S. S. Hansen, T. M. T. Larsen, M. Stevanovic, and J. M. Pedersen, "An approach for detection and family classification of malware based on behavioral analysis," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Feb. 2016, pp. 1–5.

[28] D. Ramachandram and G. W. Taylor, "Deep multimodal learning: A survey on recent advances and trends," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 96–108, Nov. 2017.

[29] X. Xu, T. Wang, Y. Yang, L. Zuo, F. Shen, and H. T. Shen, "Cross-modal attention with semantic consistence for image-text matching," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 12, pp. 5412–5425, Dec. 2020.

[30] I. J. Cruickshank and K. M. Carley, "Analysis of malware communities using multi-modal features," *IEEE Access*, vol. 8, pp. 77435–77448, 2020.

[31] P. Velickovic, D. Wang, N. D. Lane, and P. Lio, "X-CNN: Cross-modal convolutional neural networks for sparse datasets," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Dec. 2016, pp. 1–8.

[32] Y.-H. H. Tsai, S. Bai, P. P. Liang, J. Z. Kolter, L.-P. Morency, and R. Salakhutdinov, "Multimodal transformer for unaligned multimodal language sequences," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, Jul. 2019, pp. 6558–6569. [Online]. Available: <https://aclanthology.org/P19-1656>

[33] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, 1948. [Online]. Available: <http://plan9.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf>

[34] J. Kim, E.-S. Cho, and J.-Y. Paik, "Poster: Feature engineering using file layout for malware detection," in *Proc. Annu. Comput. Secur. Appl. Conf.*, Dec. 2020.

[35] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proc. EMNLP*, Aug. 2015, pp. 1412–1421. [Online]. Available: <https://aclanthology.org/D15-1166>

[36] J. Yan, G. Yan, and D. Jin, "Classifying malware represented as control flow graphs using deep graph convolutional neural network," in *Proc. 49th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2019, pp. 52–63.

[37] M. Mays, N. Drabinsky, and S. Brandle, "Feature selection for malware classification," in *Proc. MAICS*, Apr. 2017, pp. 165–170.

[38] Y. Zhang, Q. Huang, X. Ma, Z. Yang, and J. Jiang, "Using multi-features and ensemble learning method for imbalanced malware classification," in *Proc. IEEE Trustcom/BigDataSE/ISPA*, Aug. 2016, pp. 965–973.

[39] M. Ahmadi, D. Ulyanov, S. Semenov, M. Trofimov, and G. Giacinto, "Novel feature extraction, selection and fusion for effective malware family classification," in *Proc. 6th ACM Conf. Data Appl. Secur. Privacy*, Mar. 2016, pp. 183–194.

[40] R. Mitsuhashi and T. Shinagawa, "Deriving optimal deep learning models for image-based malware classification," in *Proc. 37th ACM/SIGAPP Symp. Appl. Comput.* New York, NY, USA: Association for Computing Machinery, Apr. 2022, pp. 1727–1731, doi: 10.1145/3477314.3507242.

[41] J. H. Go, T. Jan, M. Mohanty, O. P. Patel, D. Puthal, and M. Prasad, "Visualization approach for malware classification with ResNeXt," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2020, pp. 1–7.

[42] Y.-S. Liu, Y.-K. Lai, Z.-H. Wang, and H.-B. Yan, "A new learning approach to malware classification using discriminative feature extraction," *IEEE Access*, vol. 7, pp. 13015–13023, 2019.

[43] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, and S. Venkatraman, "Robust intelligent malware detection using deep learning," *IEEE Access*, vol. 7, pp. 46717–46738, 2019.



**JEONGWOO KIM** received the B.S. degree in computer science and engineering from Chungnam National University, Daejeon, in 2021, where he is currently pursuing the M.S. degree in computer science and engineering.

His research interests include malware detection, machine learning, and deep learning.



**JOON-YOUNG PAIK** received the B.S., M.S., and Ph.D. degrees in computer science and engineering from Chungnam National University, Daejeon, in 2008, 2010, and 2013, respectively.

He is currently an Associate Professor with the School of Computer Science and Technology, Tiangong University, Tianjin. His current research interests include malware detection and storage security.



**EUN-SUN CHO** (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in computer science and statistics from Seoul National University, Seoul, in 1991, 1993, and 1998, respectively.

She is currently with the Department of Computer Science and Engineering, Chungnam National University, Daejeon, where she is directing the Programming Languages and Systems Laboratory (PLAS Lab). Before she joined

Chungnam National University, in 2006, she was a Researcher with the Korea Advanced Institute of Science and Technology, Daejeon, a Visiting Professor with Ajou University, and an Assistant Professor with Chungbuk National University. Her research interests include program analysis, compilers, and computer security.

Dr. Cho is a member of the board of directors of the Korean Institute of Information Scientists and Engineers and a member of the IEEE Computer Society and ACM SIGPLAN.

...