Malware Classification

Nada Badawy, Mariam Abul-Ela

THE AMERICAN UNIVERSITY IN CAIRO

I. Abstract

Malware or "malicious software" is an umbrella term that describes any malicious program or code that is harmful to systems. malware seeks to invade, damage, or disable computers, computer systems, networks, tablets, and mobile devices, often by taking partial control over a device's operations. Thus, it is important to develop an anti-malware system that can counteract an unknown malware. the goal of this study is to predict the new malware attacks by training deep learning model to efficiently classify different 25 families of malware. To do so, we used the Malimg dataset [4] to train our models. We did 6 different experiments for 6 models (VGG16-GRU-SVM, GRU-SVM, VGG16, VGG16-GRU, deep GRU and the transformer). The results shows that deep GRU stands out among the other models with a validation accuracy of 98.7%.

II. Introduction

Malware, which is short for malicious software, is a blanket term for viruses, trojans and other harmful computer programs. It is one of the most serious security threats on the Internet today. Nowadays, our using to the software is getting bigger, and most people puts their information on different websites, and they might have sensitive information on their e-mails. In fact, most Internet problems such as spam e-mails have malware as their main cause. Computers that are compromised with malware are often networked together to form botnets, and many attacks are launched using these malicious attacker-controlled networks. They exploit target system vulnerabilities, such as a bug in legitimate software; for example, a browser or web application plugin that can be hijacked. This might lead to data theft, extortion, or the crippling of network systems. Unfortunately, detecting the malware is not that trivial. Thus, one way to predict the new attacks is to be able to differentiate between the different Malware families. Predicting the right family enables us to efficiently deal with the threat. So, our goal is to train a deep learning model to classify between 25 different families of Malware.

III. METHODOLOGY

In this section, we discuss the datasets, data preprocessing. We also discuss implementation details.

Dataset:

We mainly used the Malimg Dataset which contains 9339 malware images, belonging to 25 families/classes. We choose to use Malimg dataset because our baseline model is good at classifying images. And because it consists of images, these samples do not require a lot of pre-processing before applying image-based analysis. The Malimg dataset was created by reading the malware binaries into 8-bit

into unsigned integer composing a matrix M ∈ R m×n. This matrix may be visualized as a grayscale image having values in the range of [0, 255], with 0 representing black and 1 representing white. Table [1] summarize all class found in the Malimg dataset with number of samples in each class.

	Family/Class	Type
0	Adialer.C	Dialer
1	Agent.FYI	Backdoor
2	Allaple.A	Worm
3	Allaple.L	Worm
4	Alueron.gen!J	Worm
5	Autorun.K	Worm:AutoIT
6	C2LOP.P	Trojan
7	C2LOP.gen!g	Trojan
8	Dialplatform.B	Dialer
9	Dontovo.A	Trojan Downloader
10	Fakerean	Rogue
11	Instantaccess	Dialer
12	Lolyda.AA1	PWS
13	Lolyda.AA2	PWS
14	Lolyda.AA3	PWS
15	Lolyda.AT	PWS
16	Malex.gen!J	Trojan
17	Obfuscator.AD	Trojan Downloader
18	Rbot!gen	Backdoor
19	Skintrim.N	Trojan
20	Swizzor.gen!E	Trojan Downloader
21	Swizzor.gen!I	Trojan Downloader
22	VB.AT	Worm
23	Wintrim.BX	Trojan Downloader
24	Yuner.A	Worm

Table [1] all class in the Malimg dataset with the number of samples in each class

Data preprocessing:

Our Model requires images as input. For Malimg, we directly use the images that comprise the dataset—the only preprocessing involves separating the images into training and validation sets, resizing the images, scaling them and did some Data augmentation.

- **Separating the data:**

  We divided the data into 70% for training the model and 30% for testing it.

- **Resizing the images**:

  The images of the Malimg dataset varies in size. Thus, we resized all the images to have input shape of 224X224.

- **Data Scaling:**

  We make sure that the data was normalized in which all values are within the range of 0 and 1.

- **Data augmentation:**

  The Malimg dataset was not balanced, some class dominates others. Thus, to balance the data, we did data augmentation (horizontal flipping, vertical flipping, and adding some noise). We balanced each class to have 600 images with 15000 in total for all classes.

Implementation Details:

In this study, Google TensorFlow was used to implement the deep learning algorithms, with the aid of other scientific computing libraries: matplotlib, numpy, and scikit-learn. We conduct the training on python 3.7 environment using Jupiter notebook. The choice of TensorFlow and sklearn was influenced by the fact that they incorporate several DL best practices, including learning rate finding, stochastic gradient descent with restarts, and differential learning rates.

IV. Related Work

According to the state of art, we examined three main models for image classification to solve our main problem which is malware classification. Based on our research we found three deep neural network models.

First, VGG16 model:

VGG16, which is a version of CNN model that contains 16 deep neural network layers including 3 by 3 convolution filters. 13 of the layers are convolutional layers each three followed by Maxpooling layer. Then two cully connected layers and the output layer. Based on [3] this model achieved 92% accuracy in ImageNet dataset which is a dataset of over 14 million images

belonging to 1000 classes.

Second, GRU-SVM model:

GRU is based on RNN model which is enhanced version of LSTM which using softmax as final output layer and cross-entroby function to compute the losses. Yet GRU-SVM the parameters get learned through GRU and SVM works as classifier in a neural network architecture instead of softmax. This model reached 84% of accuracy using Malimg dataset.

Third, CNN-SVM:

The convolutional neural network CNN usually uses the softmax activation function as top layer for prediction and cross-entorpy loss function. But, according to [1] they use linear support vector machine SVM instead of softmax activation function based on the advantage that they found in its usage. SVM objective is to find the optimal hyperplane f (w, x) = w · x + b to separate two classes in a given dataset. Hence, using CNN-SVM will enhance the accuracy of CNN model. This model reach accuracy of 77% using Malimg dataset.

Fourth, Transformer:

Transformer is a model that uses attention to increase the model speed. Specifically, it uses self-attention. It contains six encoder and six decoders. Each encoder has two layers: self-attention and a feed Forward Neural Network. Also, the decoder has the same layers but between then there is an attention layer. This attention layer helps the decoder focusing on the relevant parts of the input image. Transformers have advantage over the RNN because it solves the problem of parallelization, by utilizing CNN together with attention model[5]. This model archive accuracy of 88.55% in ImageNet dataset.

VI. Experiments and Result

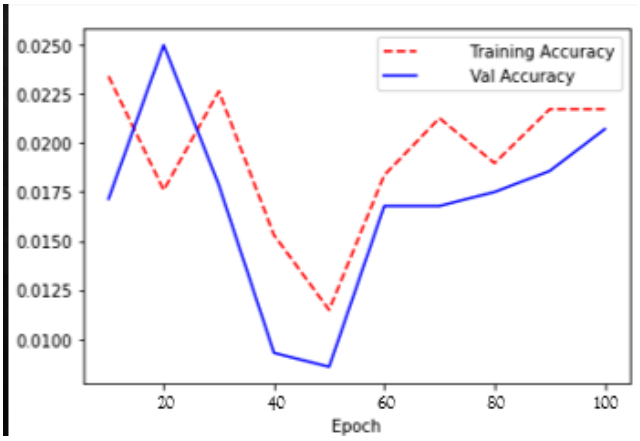
We performed a variety of experiments, on both the augmented and the original dataset. Here, we present results for 6 separate experiments, as listed in Table [2]. In the remainder of this section, we discuss each of these experiments in some detail and present the results of each.

Model name	VGG16-GRU-SVM	GRU-SVM	VGG16	VGG16-GRU	Deep GRU	Transformer
# of training data	11130 samples	11130 samples	11130 samples	11130 samples	11130 samples	11130 samples
Epochs	45	100	45	45	100	100
Batch Size	32	64	32	32	64	64
Dropout Rate	0.2	0.2	0.2	0.2	NA	0.1
Learning Rate	0.01	0.01	0.01	0.01	0.001	0.001
Total Parameters	15,3193,85	786,713	14,844,305	15,3193,85	786,713	21,759,019
Training Time	1 hour	15 min	45 min	1 hour	15 min	13 min
Validation accuracy	2%	4%	66%	54%	98.7%	91.68%

Table 2 shows the hyperparameters used to train the models.

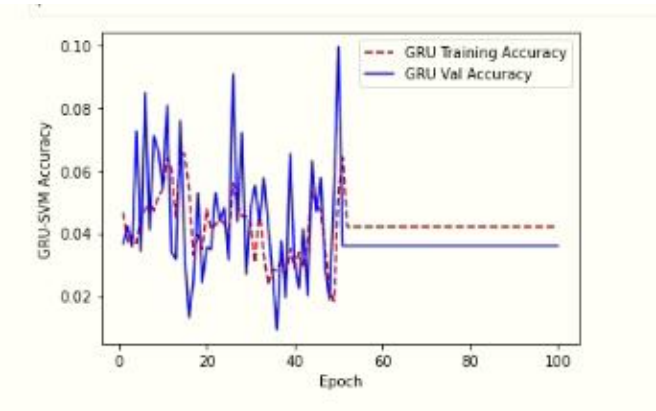
VGG16-GRU-SVM:

Our first proposed architecture was the VGG16-GRU-SVM. Table [2] shows the detailed hyperparameters used to train the model. The results show low accuracy of 2%. These results made us to try other variations.



GRU-SVM:

When we did not get good results on our proposed model (VGG16-GRU-SVM), we tried to divide it into small parts and test each part alone. So, one of these parts was the GRU-SVM. We Trained and tested the model against the hyperparameters found in table [2]. Furthermore, as shown in the results we did not get any better results.

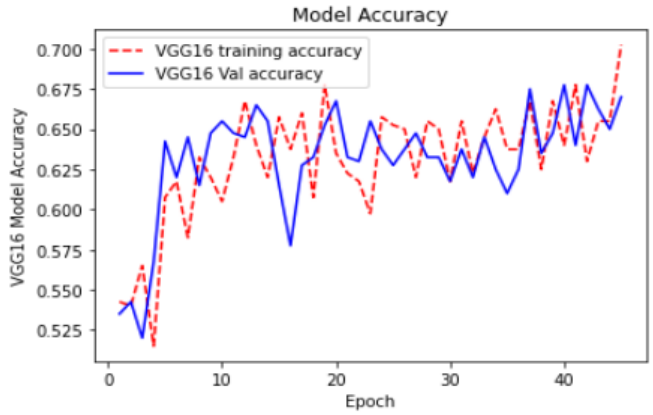


After training the GRU-SVM on 100 epochs, we get 4% validation accuracy.

VGG16:

The second part was the VGG16, so we train and test it. The results show 66% validation accuracy.

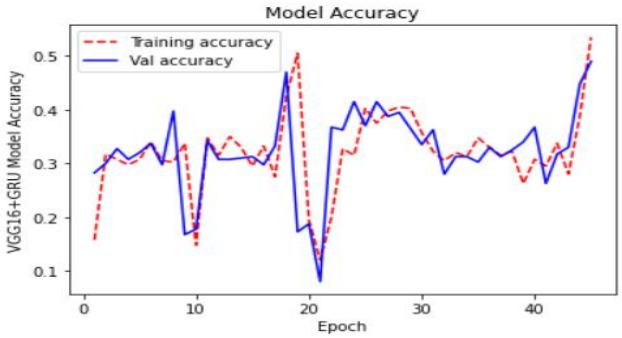
```
loss: 1.21
accuracy: 0.66
```



VGG16-GRU:

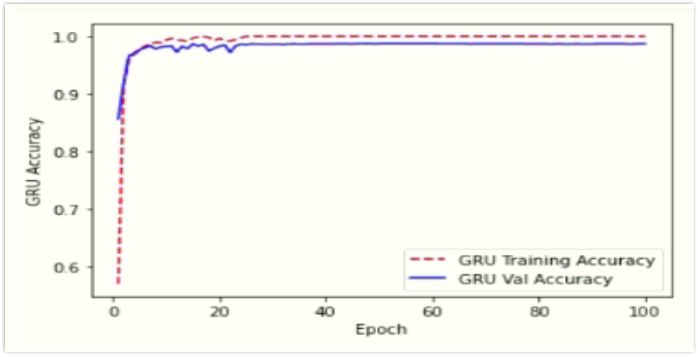
After many triers we concluded and decided that we will not include the SVM in our Model and instead we will use Softmax.

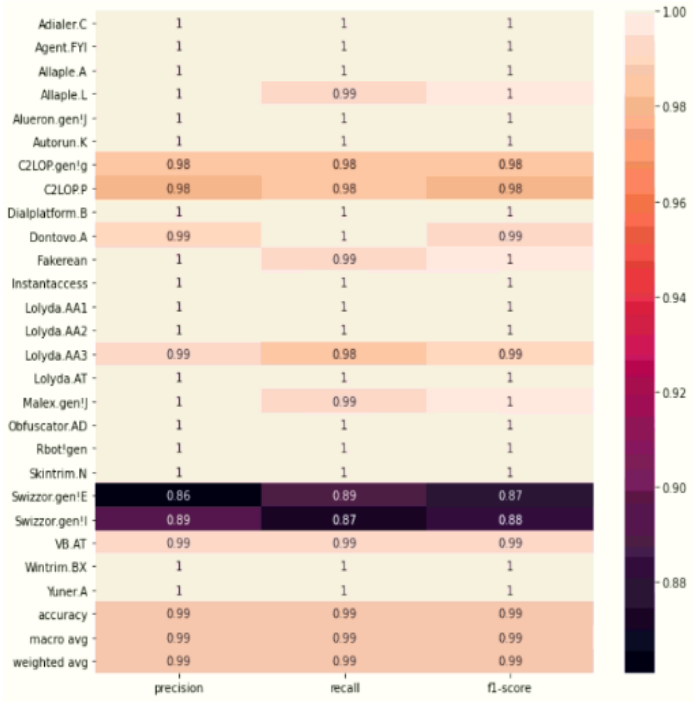
```
loss: 2.16
accuracy: 0.54
```



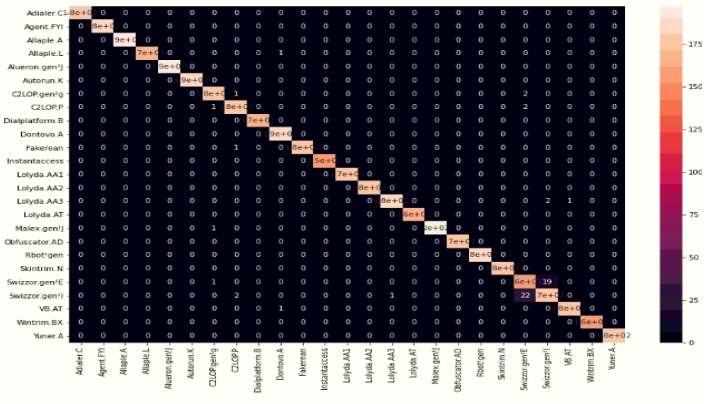
Deep GRU:

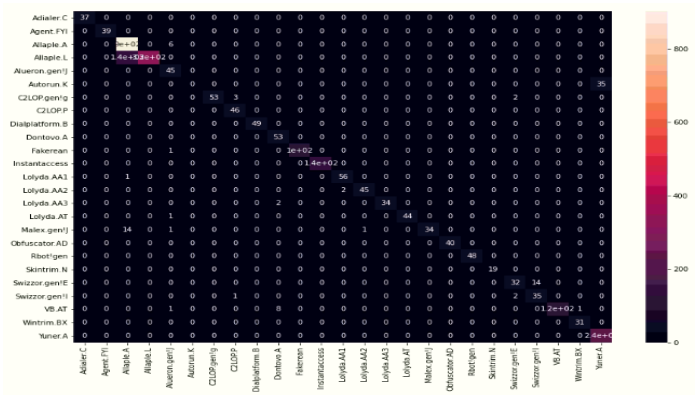
We trained our final model for 100 epochs over 11130 Malware grayscale image and tested it with 4470 samples. The model was trained to classify 25 different class of malware. The validation accuracy of the model reached 98.7% and the validation loss of 0.0915. moreover, the Confusion Matrix shows a good predictive accuracy for each malware family. However, Swizzor.gen!E and Swizzor.gen!I shows relatively low presession, recall and f1 score but in overall, all families shows high low precession, recall and f1 score Fig (1).





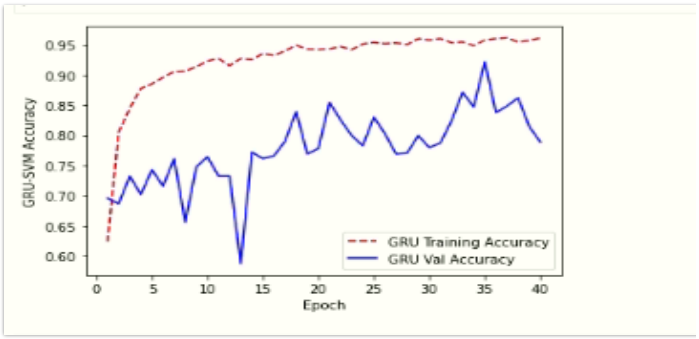
Fig(1) confusion metrics for the model

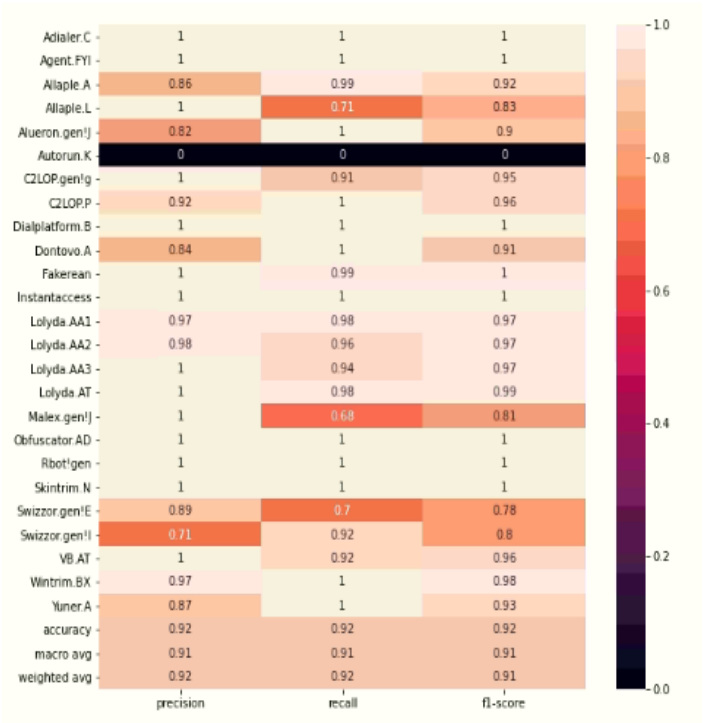




Transformer:

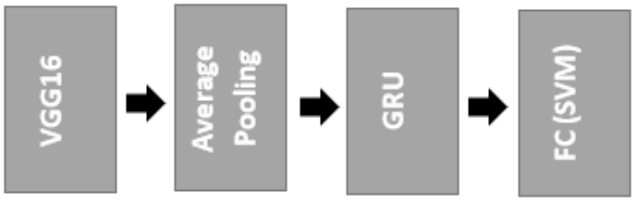
We tried the transformer to decide if we will change the deep GRU with the transformer or not. we trained it with 100 epochs and get the validation accuracy 91.68.





V. Discussion

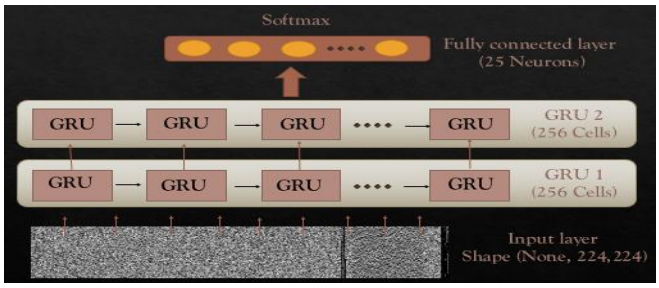
In this section we discuss the evaluation of our work from the proposed architecture (VGG16-GRU-SVM) to the final model (Deep GRU). First, After the research we made, we decided to use VGG16 as our main model adding to it the GRU as classifier and SVM instead of SoftMax activation function. As mentioned above that VGG16 has 16 layers. We used this model as pretrained model and fine tuning the last layers then add the GRU and the SVM. To add the SVM, it requires to add hinge loss as activation function with L2 regulaizar. Diagram (2) shows our proposed solution with its layered details.



Fig(2) The proposed model

Updates to the proposed Model:

After testing the models, we modified it to reach the final one that gave a satisfying result. As final model we decided to use deep GRU model with two layers with 256 units in each. The model takes input image of size (224,224) then it will go through the two GRU layers to train and finally get to the classification layer which is SoftMax with output 25 neurons corresponding to 25 classes of our dataset Malimg. Diagram (3) shows our final model architecture.



Fig(3) Final Deep GRU model

VII. Conclusion

We used the Malimg dataset prepared by [5], which consists of malware images for the purpose of malware family classification. We trained 6 models on 11130 Malware grayscale image and tested them with 4470 samples. The empirical data shows that the deep GRU model had the highest predictive accuracy among VGG16-GRU-SVM, GRU-SVM, VGG16, VGG16-GRU, and the transformer, having a test accuracy of 98.7%.

Improving the architecture design of the transformer model by adding more layers, adding better nonlinearities, and/or using an optimized dropout, may provide better insights on their application on malware classification. Such insights may reveal an information as to which architecture may serve best in the engineering of an intelligent anti-malware system.

VIII. Literature cited:

[1] Agarap, A. (2019, February 7). An Architecture Combining Convolutional Neural Network (CNN) and Support Vector Machine (SVM) for Image. doi: https://arxiv.org/pdf/1712.03541v2.pdf

[2] Agarap, A. (2019, September 10). A Neural Network Architecture Combining Gated Recurrent Unit (GRU) and Support Vector Machine (SVM) for Intrusion Detection in Network Traffic Data. Retrieved from: https://paperswithcode.com/paper/a-neural-network-architecture-combining-gated#code

[3] Gopalakrishnan, Kasthurirangan & Khaitan, S.K. & Choudhary, Alok & Agrawal, Ankit. (2017). Deep Convolutional Neural Networks with transfer learning for computer vision-based data-driven pavement distress detection. Construction and Building Materials. 157. 322-330. 10.1016/j.conbuildmat.2017.09.110.

[4] https://sarvamblog.blogspot.com/2014/08/supervised-classification-with-k-fold.html

[5] Giacaglia, G. (2020, October 5). How Transformers Work - Towards Data Science. Medium. https://towardsdatascience.com/transformers-141e32e69591