Data Augmentation with Generative Models for Improved Malware Detection: A Comparative Study*

*

Roland Burks III *, Kazi Aminul Islam†, Yan Lu‡ and Jiang Li†
†Department of Electrical & Computer Engineering, Old Dominion University, Norfolk, Virginia
{kisla001,JLi}@odu.edu
*Samford University
rburks2@samford.edu
‡Modeling, Visualization and Simulation Engineering, Old Dominion University, Norfolk, Virginia
yxxlu003@odu.edu

***Abstract*—Generative Models have been very accommodating when it comes to generating artificial data. Two of the most popular and promising models are the Generative Adversarial Network (GAN) and Variational Autoencoder (VAE) models. They both play critical roles in classification problems by generating synthetic data to train classifier more accurately. Malware detection is the process of determining whether or not software is malicious on the host's system and diagnosing what type of attack it is. Without adequate amount of training data, it makes malware detection less efficient. In this paper, we compare the two generative models to generate synthetic training data to boost the Residual Network (ResNet-18) classifier for malware detection. Experiment results show that adding synthetic malware samples generated by VAE to the training data improved the accuracy of ResNet-18 by 2% as it compared to 6% by GAN.**

***Index Terms*—Variational Autoencoders, Generative Adversarial Networks, Deep Residual Networks, Deep Learning**

I. Introduction

In todays computing landscape, the world thrives off the Internet of Things (IoT). As of now there are billions of devices that rely on the internet. This includes devices such as baby monitors, kitchen appliances, home assistants, door locks, and much more. The number of devices connected to the IoT are expected to skyrocket from 25-50 billion devices by 2020 [1] [2]. However, these numbers do not include devices such as PCs, desktops, and mobile phones. When these devices connected with internet, they are all vulnerable to malware (malicious software). This could be extremely dangerous because we do not know what the malicious users are doing with this data that they acquire. There has been much research conducted to improve the classification of malware so that the users domain is able to detect the malware in real time. The problem with detecting malware within many systems is that it requires that the end users have advanced knowledge on certain systems or use special tools. However, none of these tools can classify malware if they are modified or disguised; this also includes newly unconstrained malwares that have not been accounted for.

To deal with malware detection, many deep learning models have been proposed. Majority of the malware detection models utilized commonly implemented Convolutional Neural Networks (CNN) [3], Residual Networks (ResNet) [3], or simply a Recurrent Network [4]. These networks are widely known for image classification. However, with malware classification we must take a few extra steps to convert our raw malware bytes into images so that the deep learning model is more suitable to interpret the data. The most crucial issue with classifying malware is not necessarily the deep learning network that we decide to use, but the lack of data from certain types of malware. In order to train a deep learning model, we will need a significant amount of malware samples.

In this paper, we will compare two of the most promising deep generative models, the Generative Adversarial Network (GAN), and the Variational Autoencoder (VAE) to augment malware samples for malware detection. We will discuss the accuracy and the efficiency of both models. We have chosen these models because they are both able to generate new samples of malware that was underrepresented in typical malware data sets. There has already been research done on GAN model [5] to generate malware samples. In this paper, we will focus on generating samples with the VAE model for malware detection and compare the two models with an aim for providing practical guidance for the research community.

Some classes in the malware dataset we used have as few as 80 samples, while other classes have as many as 2,000. To properly train the classifier we generated artificial data samples with both the GAN, and VAE models so that each class has a fair number of samples in the dataset. A typical implementation of the GAN model to generate synthetic malware samples for the ResNet classifier training has been conducted by Lu *et al.* [5]. Their approach significantly improved the accuracy of classification from 84% to 90%. Our findings show that the

measurable accuracy of the VAE model implemented with the ResNet classifier only improved accuracy to 85% compared to the GAN model.

The remainder of the paper is structured as follows. We discuss related work that focus on malware classification in section II. Key terminologies that are used in the article are presented in section III. GAN and VAE models are introduced in section IV, and results are presented in section V. Following in section VI, we discuss the significance of our findings as compared to the GAN model, as well as advantages and disadvantages of both models. Finally, we conclude the paper in section VII.

II. Related Work

There have been many deep learning experiments conducted to classify malware statically. Agarap *et al.* proposed a solution for the Malimg Dataset using a support vector machine (SVM) combined with CNN, Gated Recurrent Unit (GRU), and Multi-layer perception (MLP). They achieved an accuracy of 77.22% by CNN-SVM, 84.92% by GRU-SVM, and 80.46% by MLP-SVM [6]. Lu *et al.* also attempted a similar deep learning approach with a CNN. They also added more synthetic data generated by the GAN model [7] and resulted in an 6% increase achieving an accuracy of 90% [3].

III. KEY TERMINOLOGY

A. Variational Autoencoder (VAE)

Variational Autoencoder is a popular deep generative model. Given a random variable (z) that has one dimensional distribution, we are able to generate another random variable $X = g(z)$ that has a completely diverse distribution than the original. The VAE uses 2 separate neural networks, the encoder and the decoder. The encoder takes in the input (x) and outputs an impression of (z). The decoder then attempts to reconstruct the original input(x) with a new representation of $X = g(z)$.

B. Generative Adversarial Network (GAN)

Generative Adversarial Networks are one of the most prevalent generative models although it was only introduced in 2014 by Ian Goodfellow [7]. The GAN model is based upon two different networks: a generator and a discriminator. The generator tries to fool the discriminator to thinking that it is the original input data by generating fake samples realistically like the original data. Both real data and generated data are passed into the discriminator. Therefore, the newly generated data(z) will be unique based on the input data(x) that was passed into the generator.

IV. METHODS

A. Image Representation for Malware

While dealing with malware, it is merely impossible to determine the type of attack by looking at the binary malware data. We convert the binary malware files into 8-bit gray scale images with a range of 0-255. We then convert the gray scale images into 3 channel RGB images by replicating the gray scale channels for 3 iterations.Malwares images within the same family will look very parallel in layout and texture. However, malware images from different families will be noticeably different as shown in Figure 5.

No.	Family	Family Name	No. of Variants
1	Worm	Allaple.L	1591
2	Worm	Allaple.A	2949
3	Worm	Yuner.A	800
4	PWS	Lolyda.AA 1	213
5	PWS	Lolyda.AA 2	184
6	PWS	Lolyda.AA 3	123
7	Trojan	C2Lop.P	146
8	Trojan	C2Lop.gen!G	200
9	Dialer	Instantaccess	431
10	Trojan Downloader	Swizzor.gen!I	132
11	Trojan Downloader	Swizzor.gen!E	128
12	Worm	VB.AT	408
13	Rogue	Fakerean	381
14	Trojan	Alueron.gen!J	198
15	Trojan	Malex.gen!J	136
16	PWS	Lolyda.AT	159
17	Dialer	Adialer.C	125
18	Trojan Downloader	Wintrim.BX	97
19	Dialer	Dialplatform.B	177
20	Trojan Downloader	Dontovo.A	162
21	Trojan Downloader	Obfuscator.AD	142
22	Backdoor	Agent.FYI	116
23	Worm:AutoIT	Autorun.K	106
24	Backdoor	Rbot!gen	158
25	Trojan	Skintrim.N	80

Fig. 1. Malimg Dataset

B. Malimg Data Set

For our experiment, we have chosen to use the Malimg data set. The Malimg data set contains 25 malware families (classes), while each family has a varying quantity of samples. The Allaple.A family has 2949 samples while the Skintrim.N family has only 80. The remaining 23 class samples fluctuate between these two ranges are shown in Figure 1.

C. Deep Residual Networks

Deep learning models have been extremely reliable and promising in image classification. However, there are pros and cons to deep learning networks. While training most deep learning models, as more layers are added to the network, the gradients of the loss function approaches to 0, which makes the network more difficult to train. This is referred to as the gradient vanishing problem. He $et.al$ formed a residual learning framework that makes training deep learning networks easier, which offered reformulates the layers as learning residual functions with reference to the layers input [5]. This is by far one of the most useful computer vision and image classification models. There is plenty of evidence that the deep residual network has been controlled and modified to solve many important classification problems.

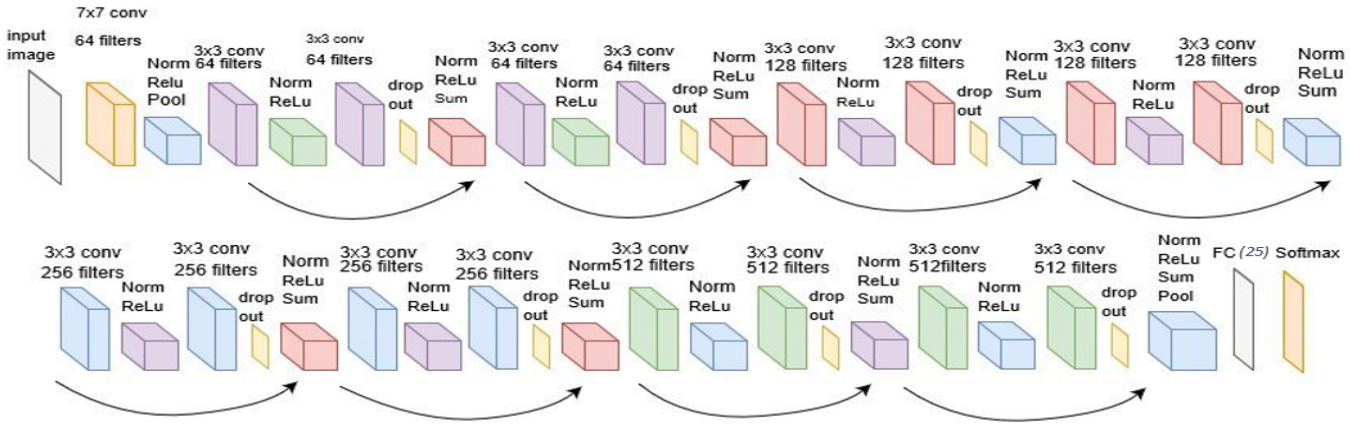


Fig. 2. Deep Residual Network

[5]

We utilized one configuration of deep residual network, ResNet-18, for malware detection in this study. ResNet-18 consists of 17 convolutional layers of different convolutional kernel sizes, a normalization layer, a linear unit layer, a pooling layer, and a SoftMax layer for classificaiton. A common issue with multi-layered networks is overfitting. To resolve this problem, we added drop-out layers between convolutional and normalization layers as shown in Figure 2 [5].

D. Variational Autoencoder

VAE model is the predecessor of the famous GAN model. A modern VAE architecture consists of 3 major components: an encoder, a decoder and a loss function. Both the encoder and decoder are convolutional neural networks. Encoder compresses an original image to a low-dimensional vector, z, while decoder reconstruct the original image based on z. After training, the decoder can be used to generated more data given a random variable z. VAE is capable of synthesizing new data, $Z = g(z)$, by sampling in the low-dimensional space z as shown in Figure 3.

Figure 3 shows the details of the VAE model used in this study. The encoder consists of five layers of which the first four layers are convolutional layer followed by a dense layer. The first convolutional layer contains 128 convolutional kernels of size 3x3 and a relu activation function. The second convolutional layer also contains 128 layers, a kernel size of 2x2, and has a stride of 2x2. The third convolutional layer has 64 filters, a kernel size of 3x3, a relu activation as the previous layer, and a stride of 2x2. The fourth layer consists of 32 filters, and a kernel size of 3x3, a relu activation and another stride of 2x2. The final dense layer flattens the data and passes it on to the decoder for up-sampling.

The first layer in the decoder is the up-sampling layer which contains 64 filters. The second layer is a de-convolutional layer which contains 32 filters, a kernel size of 5x5 ,a stride of 1, and a relu activation. The third layer is a de-covolutional layer that consits of 64 filters, a kernel size of 3x3, a stride of 1, and a relu activation The fourth layer is also a de-convolutional layer which contains 128 filters, a filter size of 5x5, a stride of 1, and a relu activation. The final de-convolutinal layer contains 128 filter, a kernel size of 3x3, a stride of 1, and a sigmoid activation.

E. Conditional VAE Generated Malware Samples

Figure 3 shows the architecture of VAE model we used. Initially, we pass the original malware sample as input. The input data is encoded using the deep encoder network. The VAE model uses a decoder network for reconstruction The decoder network reconstruct the sample from the latent variable, z to match the input data. If the reconstructed sample does not match the input sample it produces a high loss. Using this loss, the VAE network optimizes both encoder and decoder network. We trained the VAE model for 150 epochs in order for us to generate artificial malware samples and to fairly compare the results by Lu *et al.* where GAN was used to generate malware samples [5]. Initially , the samples in the Malimg Dataset are all converted to grey-scale images with dimensions of 32x32.

F. Training the Classifier

Using the VAE model ,2000 malware samples were generated for the training. We began training the classifier by only using real training data from the Malimg Dataset. We initially took the first 30 samples of each class as the testing data and used the remaining as the training data. The residual network only resulted in an accuracy of 83%.

At this point we randomly selected 100 random VAE samples from the 2000 generated samples for each class so that we are able to retrain and retest the Residual Network classifier. To prevent over-fitting the model we used the Keras API early stopping, so that we could stop the training once the model performance stopped improving. Then we were able to train the 18-layered ResNet classifier by implementing the newly generated artificial data. For each iteration we trained

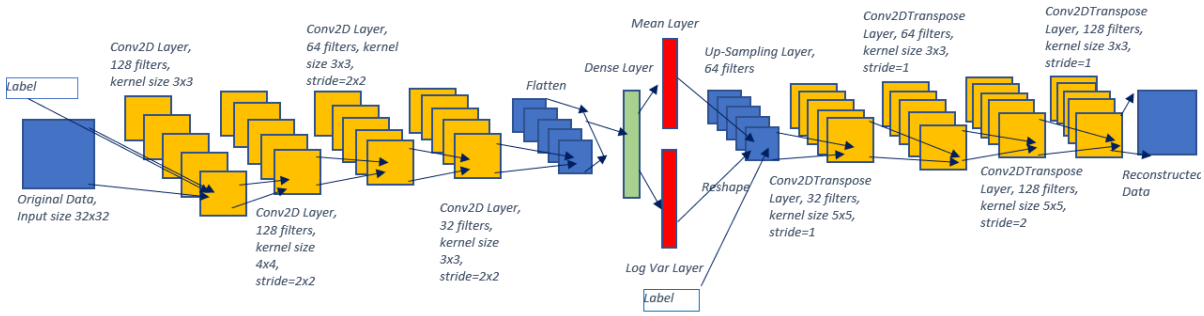
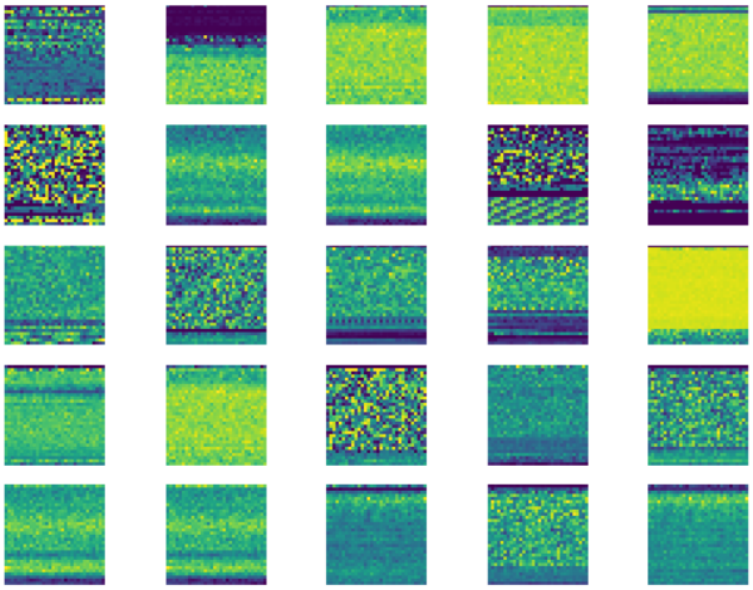


Fig. 3. Variational Autoencoder



Group 1 of Generated Samples

Fig. 4. VAE Generated Sample Images

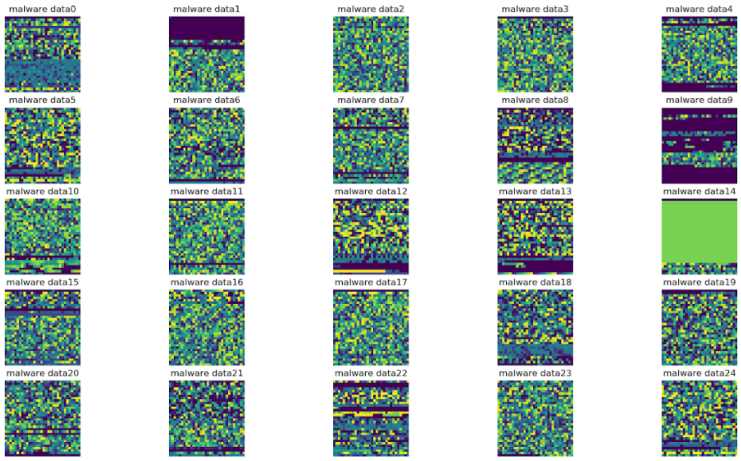


Fig. 5. Original Malware Sample Images

the model for 150 epochs. We conducted this testing on a Lenovo desktop that runs an Intel Core i3-7100 CPU. It took roughly 2 minutes for each epoch with a batch size of 250.

G. Performance Metric

During this experiment we concluded our results with precision, recall, and f1-scores to evaluate the optimization of the deep residual network. For each class we used the precision to show how accurate the model made predictions. We then used the recall to calculate the number of true positives the model encountered by categorizing it as a positive. Finally, we use the f1-score to view the balance between precision and recall. The formulas are shown below.

$$Precision = \frac{true\ positive}{true\ positive + false\ positive} \tag{1}$$

$$Recall = \frac{true\ positive}{true\ positive + false\ negative} \tag{2}$$

$$f1 - score = \frac{precision * recall}{precision + recall} \tag{3}$$

V. Results

A. Testing Accuracy

As described earlier in the Training the Classifier section we only trained the classifier with the real training data from the Malimg Dataset. The classifier resulted in an accuracy of 83% as shown in Figure 6.

By using the artificial generated malware samples that were produced by the VAE model the precision, recall, and f1-score are only slightly improved compared to Lu *et al.* GAN model [5]. Both the VAE and the GAN model served the same purpose for generating synthetic malware samples, however we can imply that Lu *et al.* GAN model showed a significant performance increase as shown in Figure 7 [5].

Both the GAN and VAE model performed similarly when it came to generate artificial data to feed into the same residual network classifier. However, the GAN resulted in better accuracy. They both shared very similar complexities when it comes to performance. Compared to Lu *et al.* GAN model our VAE only showed a slight 2% increase compared to Lu *et al* 6% increase as shown in Figure 7.

	precision	recall	f1-score	support
Allaple.L	1.00	1.00	1.00	30
Allaple.A	1.00	1.00	1.00	30
Yuner.A	0.35	0.47	0.40	30
Lolyda.AA 1	0.63	0.80	0.71	30
Lolyda.AA 2	0.88	1.00	0.94	30
Lolyda.AA 3	1.00	1.00	1.00	30
C2Lop.p	0.45	0.50	0.48	30
C2Lop.gen!G	0.35	0.40	0.38	30
Instantaccess	1.00	0.93	0.97	30
Swizzor.gen!I	1.00	1.00	1.00	30
Swizzor.gen!E	1.00	0.93	0.97	30
VB.AT	1.00	1.00	1.00	30
Fakerean	0.97	1.00	0.98	30
Alueron,gen!J	0.94	0.97	0.95	30
Malex.gen!J	1.00	1.00	1.00	30
Lolyda.AT	0.90	0.93	0.92	30
Adialer.C	0.58	0.50	0.54	30
Wintrim.BX	1.00	1.00	1.00	30
Dialplatform.B	0.91	0.97	0.94	30
Dontovo.A	1.00	1.00	1.00	30
Obfuscator.AD	0.47	0.27	0.34	30
Agent.FYI	0.43	0.33	0.38	30
Autorun.K	1.00	1.00	1.00	30
Rbot!gen	0.92	0.73	0.81	30
Skintrim.N	1.00	1.00	1.00	30
micro avg	0.83	0.83	0.83	750
marco avg	0.83	0.83	0.83	750
weighted avg	0.83	0.83	0.83	750
samples avg	0.83	0.83	0.83	750

Fig. 6. Real Training Data Accuracy

VI. DISCUSSION

The significance of generating artificial malware samples using the Variational Autoencoder showed that it improves the accuracy once implemented into the Residual Network. However, using the same Residual Network, the Generative Adversarial Networks generated synthetic samples improved the accuracy by a much greater margin than the Variational Autoencoder. Each generative model used the same parameters that were required to generate the artificial data to ensure that the comparison was fair. They both rely on two separate networks. The GAN takes a gaming approach where one network (generator) attempts to confuse the network(discriminator) by generating artificial training data. Whereas the VAEs encoder receives input data and configures a hidden latent representation of the input data. The decoder network learns to output the original data from the given encoded data. This illustrates that the efficiencies of the two models differ. However, both models do fix the issue of small data samples.

Both models have their advantages and disadvantages. Initially both models have proven to be successful of generating any type of data, which is extremely beneficial in a world that thrives from big data. The GAN can perform unsupervised learning, while the VAE model supports semi-supervised and unsupervised learning [8]. A disadvantage of the GAN model is there is no clear representation of x (real data), and that the discriminator must but in sync with the generator during the training phase so the generator does not collapse too many values of z to where it will match x [7]. A disadvantage of VAE is that it often produces distorted reconstruction due to too much noise. This results in the generated samples coming from the VAE being less valuable than those coming from

	precision	recall	f1-score	support
Allaple.L	1.00/1.00	1.00/1.00	1.00/1.00	30
Allaple.A	1.00/1.00	1.00/1.00	1.00/1.00	30
Yuner.A	0.36/0.62	0.50/0.60	0.42/0.61	30
Lolyda.AA 1	0.64/0.65	0.77/0.80	0.70/0.72	30
Lolyda.AA 2	0.77/1.00	1.00/1.00	0.87/1.00	30
Lolyda.AA 3	1.00/1.00	1.00/1.00	1.00/1.00	30
C2Lop.p	0.48/0.52	0.67/0.53	0.56/0.52	30
C2Lop.gen!G	0.55/0.61	0.57/0.63	0.56/0.62	30
Instantaccess	1.00/1.00	0.93/0.97	0.97/0.98	30
Swizzor.gen!I	1.00/1.00	1.00/1.00	1.00/1.00	30
Swizzor.gen!E	0.97/0.97	0.97/0.97	0.97/0.97	30
VB.AT	1.00/1.00	1.00/1.00	1.00/1.00	30
Fakerean	0.97/0.97	1.00/1.00	0.98/1.00	30
Alueron,gen!J	1.00/1.00	0.97/1.00	0.98/0.98	30
Malex.gen!J	1.00/1.00	1.00/1.00	1.00/1.00	30
Lolyda.AT	0.96/1.00	0.90/0.97	0.93/0.98	30
Adialer.C	0.69/0.81	0.60/0.83	0.64/0.82	30
Wintrim.BX	1.00/1.00	1.00/1.00	1.00/1.00	30
Dialplatform.B	0.97/0.93	0.97/0.90	0.97/0.92	30
Dontovo.A	1.00/0.97	1.00/1.00	1.00/0.98	30
Obfuscator.AD	0.70/0.81	0.23/0.73	0.35/0.77	30
Agent.FYI	0.50/0.72	0.40/0.70	0.44/0.71	30
Autorun.K	1.00/0.97	1.00/1.00	1.00/0.98	30
Rbot!gen	0.88/1.00	0.70/0.83	0.78/0.91	30
Skintrim.N	1.00/1.00	1.00/1.00	1.00/1.00	30
micro avg	0.85/0.90	0.85/0.90	0.85/0.90	750
marco avg	0.86/0.90	0.85/0.90	0.84/0.90	750
weighted avg	0.86/0.90	0.85/0.90	0.84/0.90	750
samples avg	0.85/0.90	0.85/0.90	0.85/0.90	750

Fig. 7. VAE Results Compared to GAN

GAN.

The importance of this comparative study is to show how similar constructed models produce the same type of data with different efficiencies. The under-representation of data within certain data set can heavily impact the results of your study. While dealing with the Malimg Dataset it was not reasonable to train the Residual Network to classify different types of malware attacks if the margins of different malware samples were too large.

The computational efficiency for each model varied slightly. It took 45 minutes to train the VAE model for 50 epochs in order to generate artificial data. When the generated VAE samples were added to the Residual Network it took 1 hour to declare an accuracy report.

Although both models proved to be useful in generating malware samples to improve the accuracy of the Residual Network, we recommend using the GAN model to generate training data for the classifier. The results will be significantly higher compared to those of the VAE model. The GAN reduces

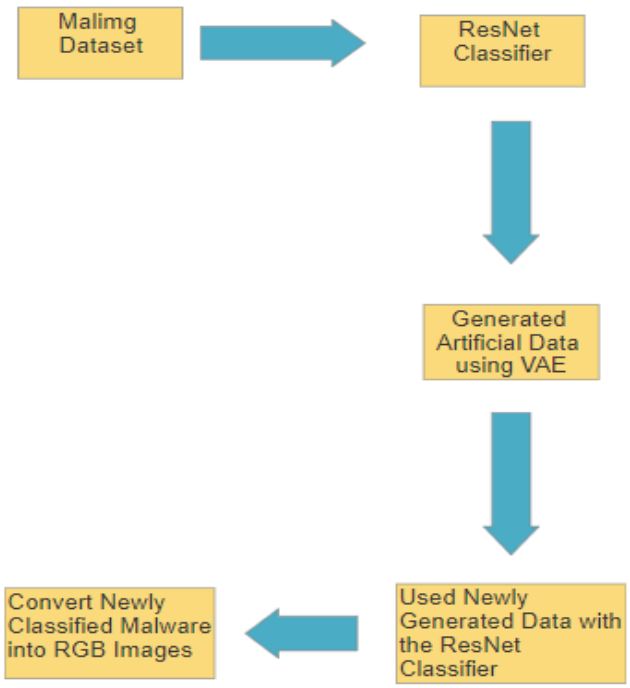


Fig. 8. Proposed Model Architecture

the concern of distortion by limiting feature lost so that the training process for the classifier is more efficient.

VII. CONCLUSION

We proposed a Variational Autoencoder model to resolve the problem of under-representation of data by generating artificial malware data. Our results determined that by inserting the VAE generated samples into our Residual Network, the performance increased by 2%. This model was compared to the famous Generative Adversarial Network model, which resulted in a promising 6% increase.

Primarily, we trained the Residual Network without adding the generated malware samples. The accuracy for the Residual Network resulted in 83%. After adding the artificial generated data from the VAE, the results increased to 85%, whereas the results from the GAN increased to 90% [5]. We also exposed the advantages and disadvantages of both the VAE and GAN models.

We were also able to convert the VAE generated samples bytecodes in RGB images to better improve our deep learning classifications. Generative Networks reduced the expertise needed to within certain systems to detect various malware attacks. It also limits the professional tools needed.

After carefully examining the performance and results of both generative networks, we have concluded that using the GAN model to generate artificial malware data is more efficient than the VAE model when it comes to deep learning malware classification.

REFERENCES

[1] I. Lee and K. Lee, "The internet of things (iot): Applications, investments, and challenges for enterprises," *Business Horizons*, vol. 58, no. 4, pp. 431–440, 2015.

[2] B. D. Weinberg, G. R. Milne, Y. G. Andonova, and F. M. Hajjat, "Internet of things: Convenience vs. privacy and secrecy," *Business Horizons*, vol. 58, no. 6, pp. 615–624, 2015.

[3] S. Yue, "Imbalanced malware images classification: a cnn based approach," *arXiv preprint arXiv:1708.08042*, 2017.

[4] R. Pascanu, J. W. Stokes, H. Sanossian, M. Marinescu, and A. Thomas, "Malware classification with recurrent networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 1916–1920.

[5] J. L. Yan Lu, "Generative adversarial network for improving deep learning based malware classification," *Proceeding of 2019 Winter Simulation Conference*, 2019.

[6] A. F. Agarap and F. J. H. Pepito, "Towards building an intelligent anti-malware system: a deep learning approach using support vector machine (svm) for malware classification," *arXiv preprint arXiv:1801.00318*, 2017.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[8] J. T. Chauhan, "Comparative study of gan and vae," *International Journal of Computer Applications*, vol. 975, p. 8887.