CISSE

Efficient Machine Learning for Malware Detection

Thomas A. Koch

Dr. Tamirat Abegaz, GASF, GCTI, GCFA, GCIH

Dr. Hyungbae Park, GCFE, GREM, GCLD, GMLE

UNG

UNIVERSITY of NORTH GEORGIA

THE MILITARY COLLEGE OF GEORGIA

Introduction

Malware Detection is a Critical aspect of Cybersecurity

Ransomware alone – 26 Million monthly attacks in 2024

SonicWall threat research – 6 Billion malware hits in 2024

Why Machine Learning for Malware Detection?

Signature-Based Detection vs. Machine-Learning

- Signature-Based Detection:
  - Relies on known malware signatures
  - Susceptible to zero-day malware
  - Relies on an external database
  - Needs updates
  - Can be predictable

- Machine-Learning Approach:
  - Effective against novel malware
  - Does not rely on a database
  - Scalable
  - Harder to predict

Artificial Intelligence & Machine Learning , Next-Generation Technologies & Secure Development

Google AI Agent Finds Zero-Day in Popular Database Engine

Now-Fixed Flaw Is Big Sleep's First Real-World Bug Find, Say Researchers

Rashmi Ramesh (rashmiramesh_) • November 5, 2024

Convolutional Neural Networks (CNN)

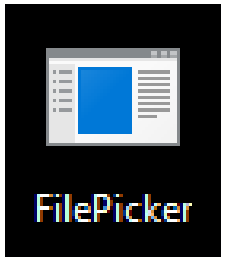
- Deep learning AI neural networks, very effective for complex datasets

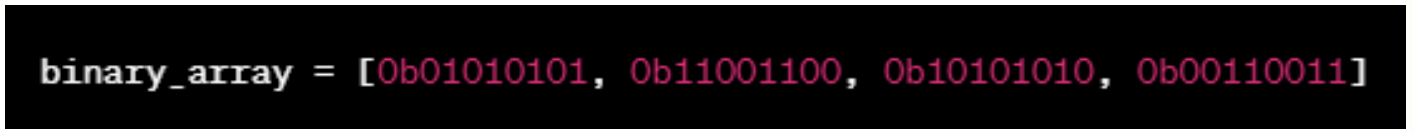
- **Research question:**
  - To what extent could a Convolutional Neural Network differentiate between benign and malicious Windows executables, and how accurate is it in predicting harmful Windows binaries?

  - How might it compare to standard signature-based detection?

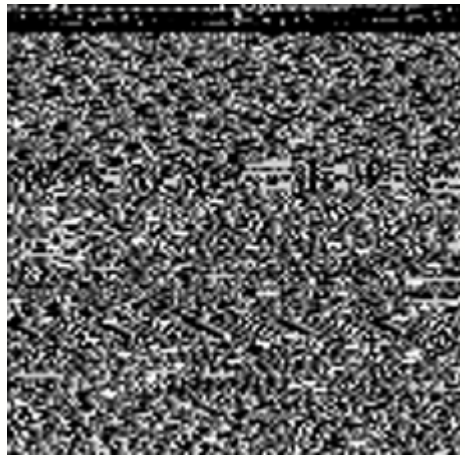
Our Approach

- Converted binaries to grayscale images
- Trained multiple CNN models on these images





- **Implications:**
  - Enhanced detection of previously unknown malware
  - Resource efficient and scalable solution



Methodology

- Gather a Dataset
  - 200 malicious files
  - 200 benign executables
- Preprocess the data
  - Trimming and padding
  - Convert to grayscale image
- Build a CNN model
  - TensorFlow Keras API with Python
  - Local WSL 2 environment
- Train the CNN
  - Initial expected accuracy above 80%
  - 70-79% - less than optimal
  - <= 69% - considered not viable
- Evaluate accuracy
  - Unseen test set of 100 programs
  - Compared to Windows Defender

Model Architecture

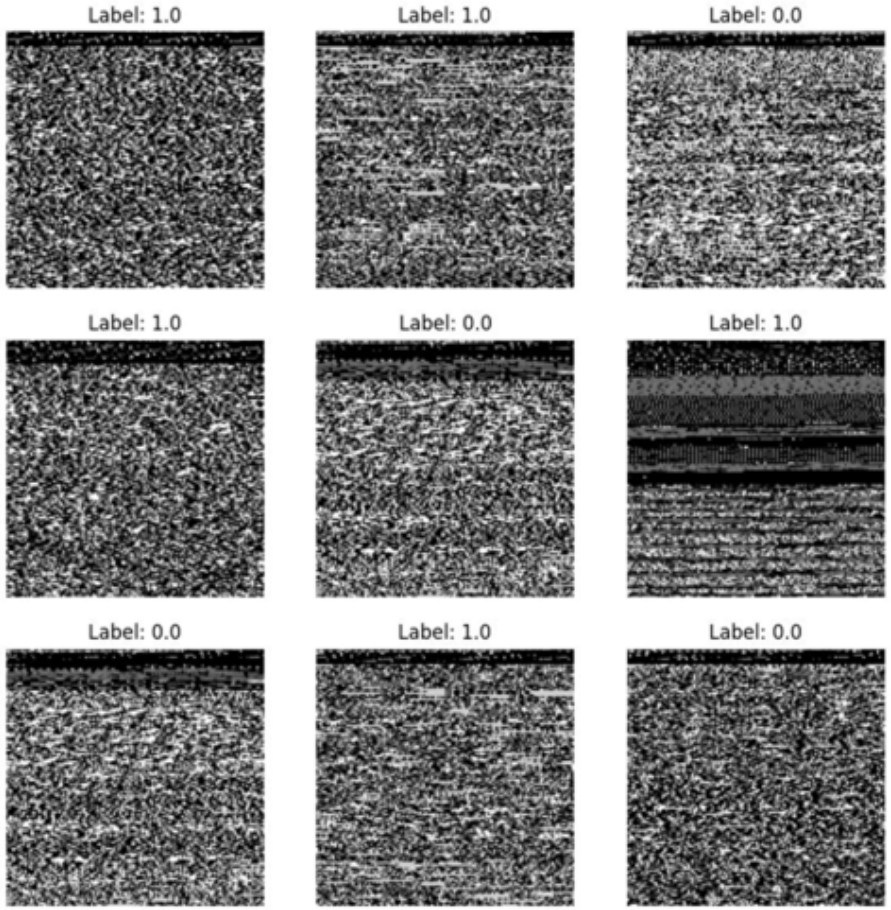
```
optimizer = Adam(learning_rate=custom_learning_rate)

model = models.Sequential()
model.add(layers.DepthwiseConv2D(kernel_size=(3, 3), activation='relu', kernel_regularizer=l2(0.01), padding='same', input_shape=(128, 128, 3)))
model.add(layers.DepthwiseConv2D(kernel_size=(3, 3), activation='relu', kernel_regularizer=l2(0.01), padding='same', input_shape=(128, 128, 3)))
model.add(layers.Conv2D(10, (3, 3), activation='relu', groups=1, input_shape=(128, 128, 3)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(Dropout(0.5))
model.add(layers.Conv2D(10, (3, 3), activation='relu', groups=1, input_shape=(128, 128, 3)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(Dropout(0.5))
model.add(layers.Dense(128, activation='relu'))
#model.add(Dropout(0.5)) - this layer was frequently included and excluded to analyze overfitting
model.add(layers.Dense(1, activation='sigmoid'))
model.compile(optimizer=optimizer, loss='binary_crossentropy', metrics=['accuracy'])

model.fit(trainGenerator, epochs=num_epochs, validation_data=testGenerator, callbacks=[tensorboard_callback])
```

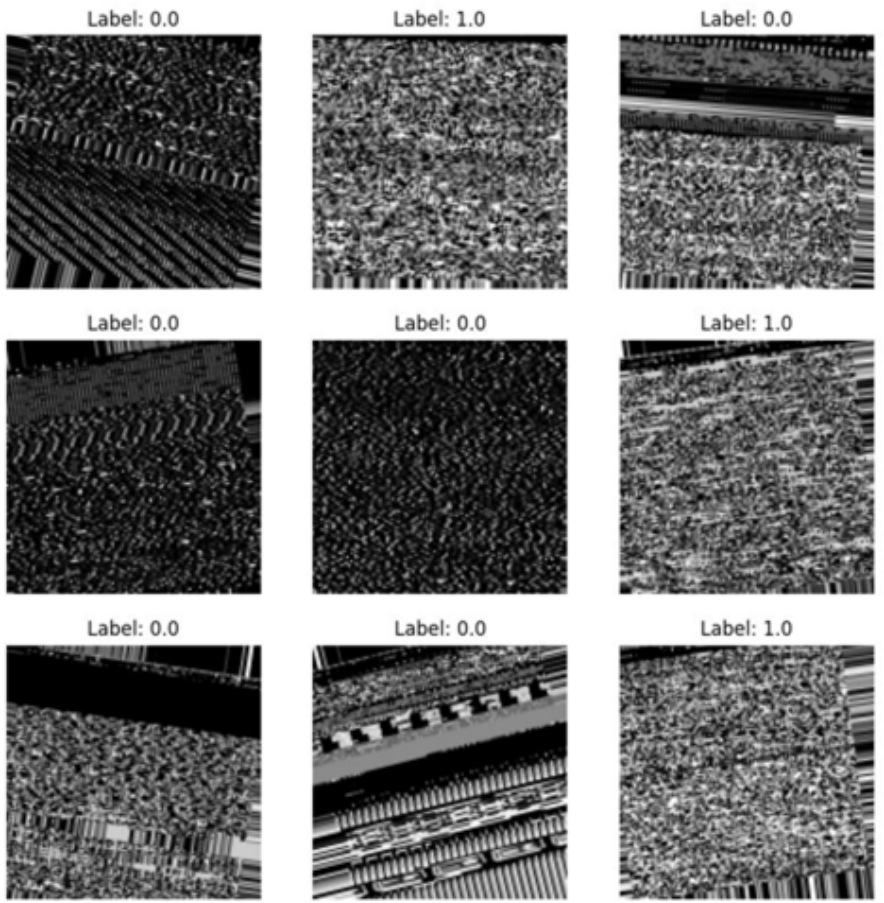
Data Augmentation

```
trainDatagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=2,
    width_shift_range=0.01,
    height_shift_range=0.01,
    shear_range=0.01,
    zoom_range=0.01,
    horizontal_flip=True,
```

UNG | UNIVERSITY of NORTH GEORGIA™

Augmentation Visualization





Evaluation and Results

Result Metrics

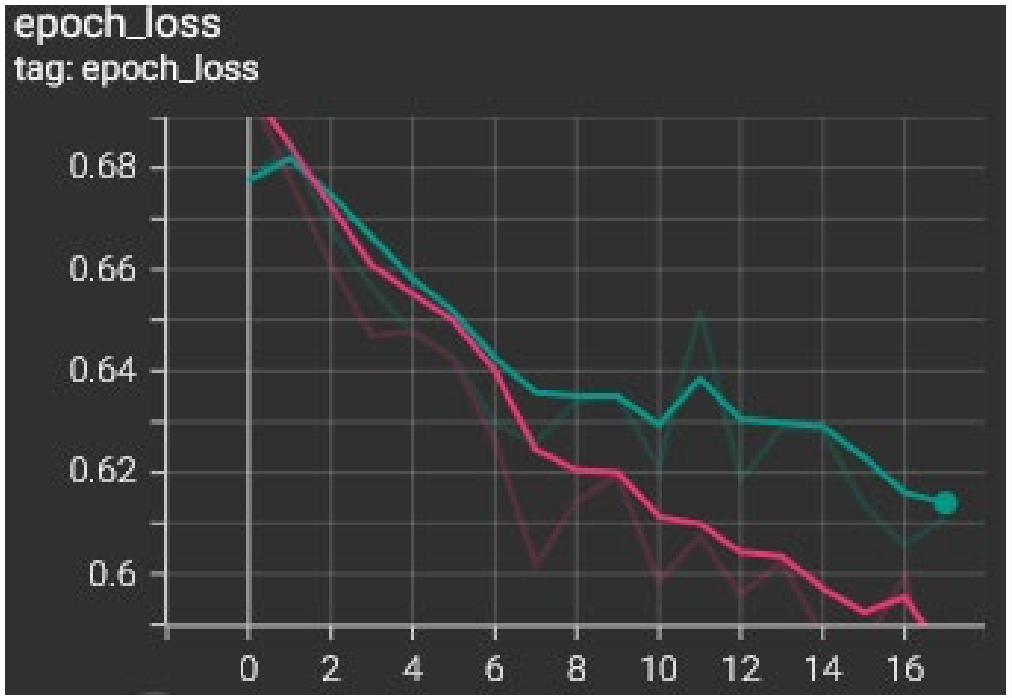
- Accuracy vs. Validation accuracy
- Loss graphs
- 80/20 split for training data

- Initial dataset of 400 gathered successfully
- Benign files sourced from Windows apps
- Malicious files sourced from GitHub malware repositories and MalwareBazaar

First Run Results

- Accuracy: 70%
- Validation Accuracy: 67.5%

- Decided to increase dataset size to 800 images

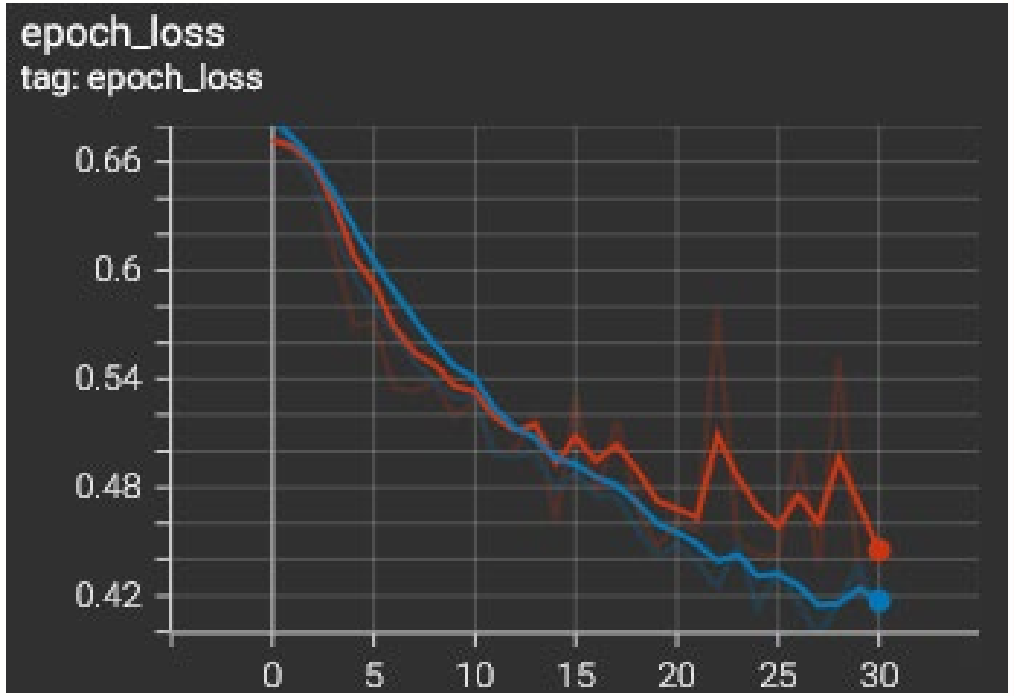


UNG | UNIVERSITY of NORTH GEORGIA™

Second Dataset Results

- Accuracy: 80.2%
- Validation Accuracy: 78.5%

- Due to the jump, we increase the dataset again to 3,000 images

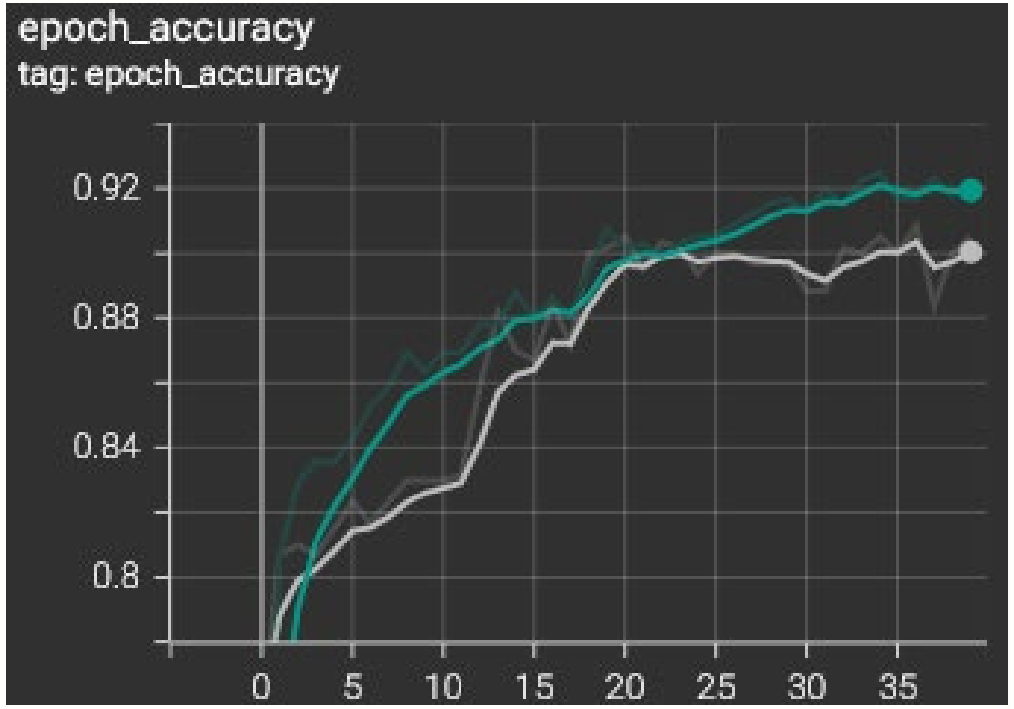


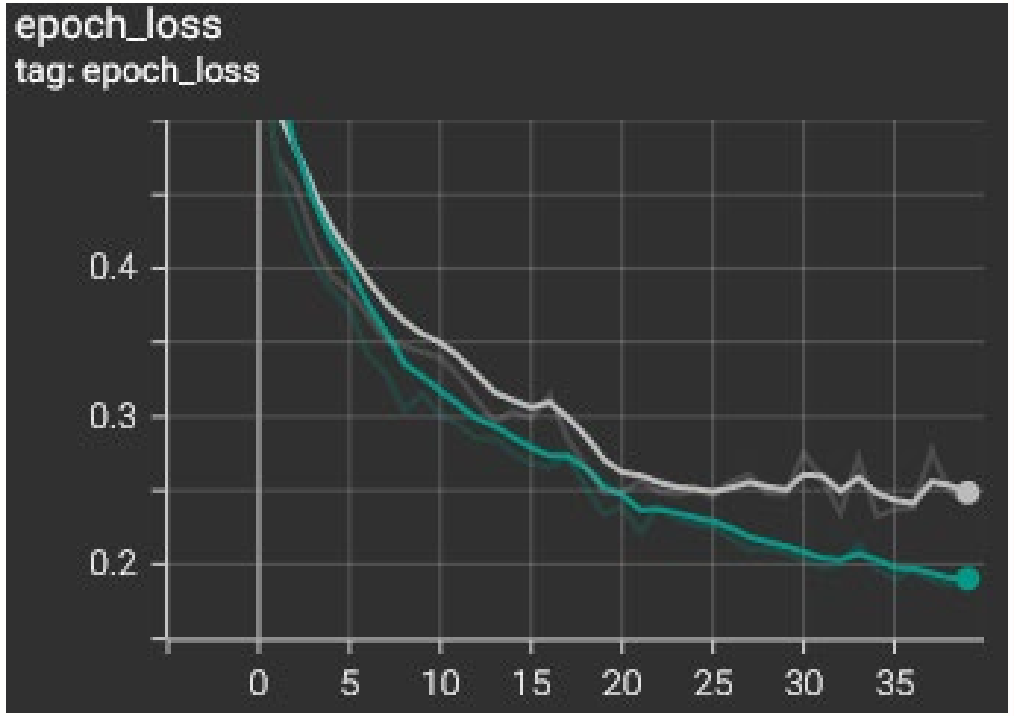
UNG | UNIVERSITY of NORTH GEORGIA™

Third Dataset Results

- Accuracy: 91%
- Validation Accuracy: 90%
- Training Loss .189
- Validation Loss .240

- Final dataset increase: 4,188 images. Slight imbalance in favor of malware images

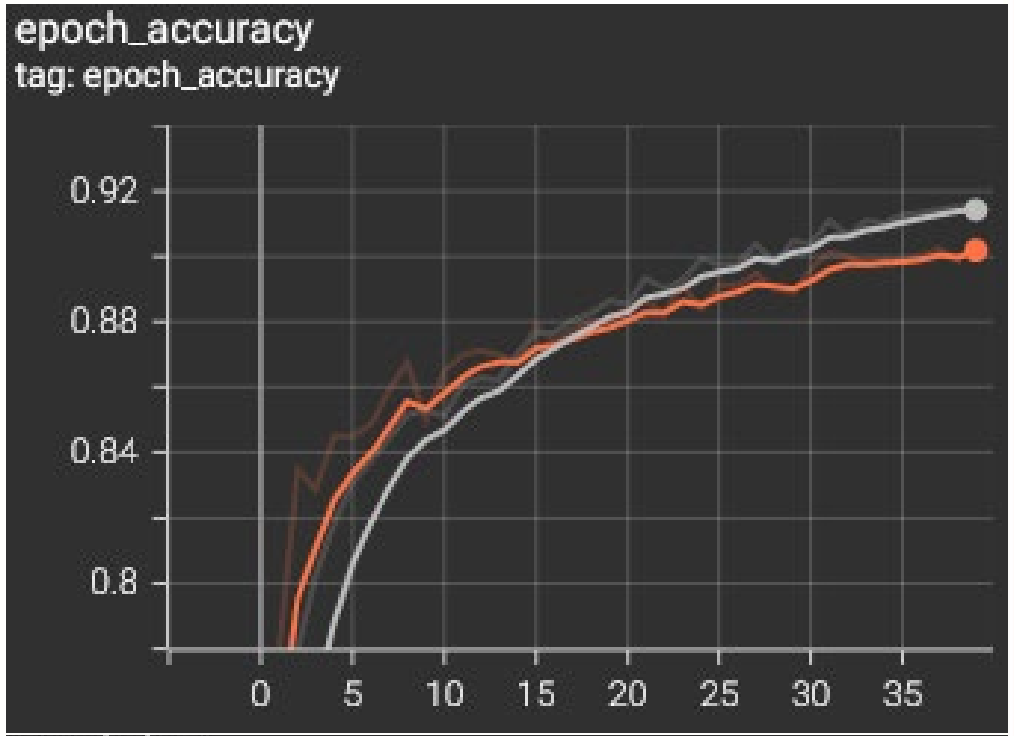


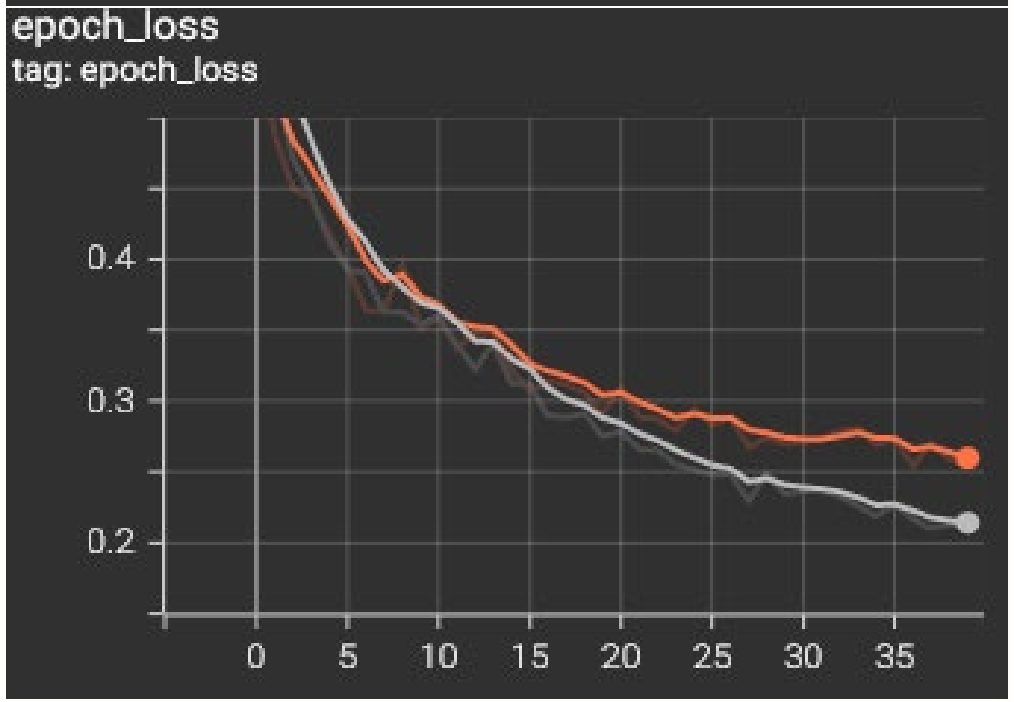


UNG | UNIVERSITY of NORTH GEORGIA™

Final Dataset Results (Model 1)

- Accuracy: 91%
- Validation Accuracy: 90%
- Training Loss .211
- Validation Loss .253





UNG | UNIVERSITY of NORTH GEORGIA™

GPT-4 Trained Model

- Simpler architecture, fewer nodes
- Automated early stopping
- Dynamic learning rate adjustments

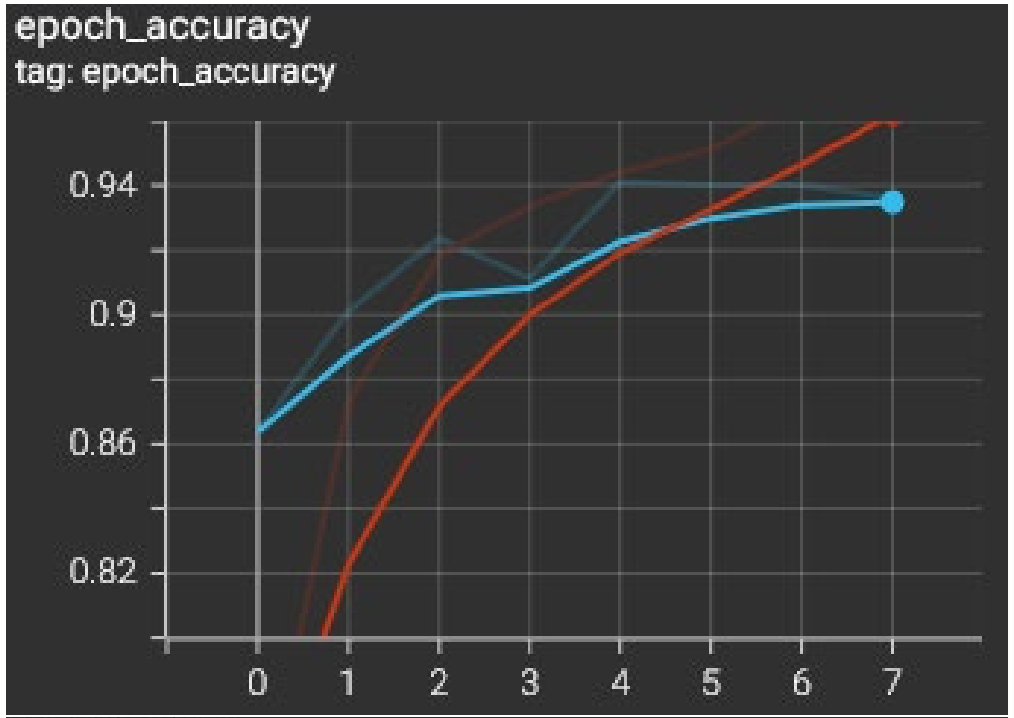
```
# Model Building
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(128, 128, 1)),
    MaxPooling2D(2, 2),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),
    Flatten(),
    Dense(64, activation='relu'),
    Dropout(0.5),
    Dense(1, activation='sigmoid')
])
```

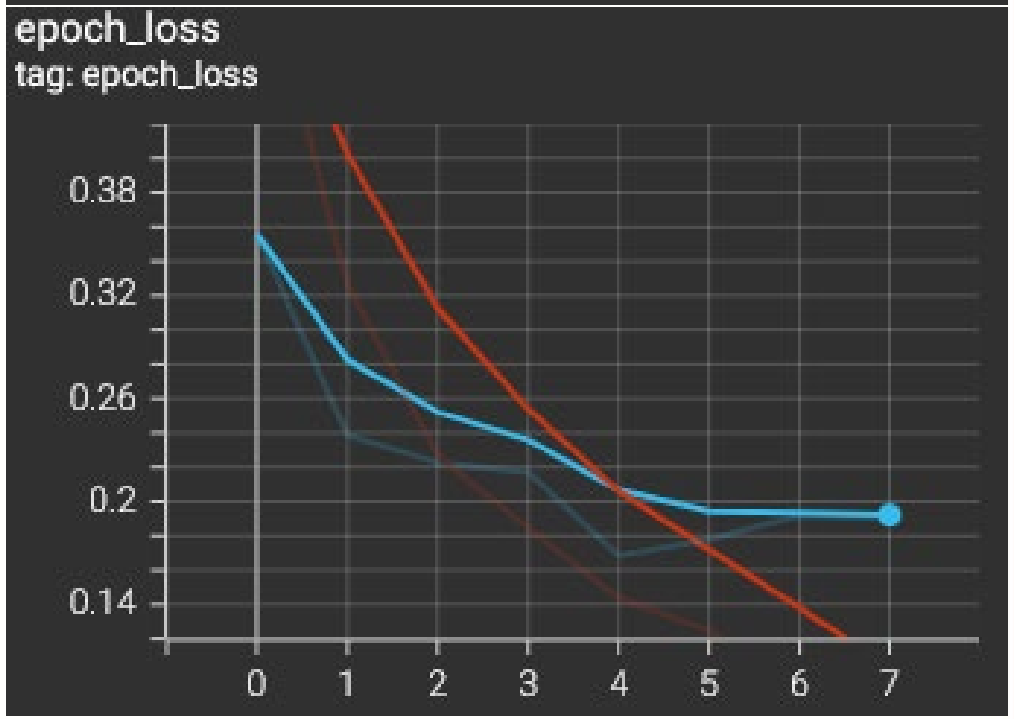
```
# Callbacks
early_stopping = EarlyStopping(
    monitor='val_loss',
    patience=3,
    verbose=1,
    restore_best_weights=True
)
```

```
reduce_lr = ReduceLROnPlateau(
    monitor='val_loss',
    factor=0.2,
    patience=2,
    verbose=1,
    min_lr=0.00001
)
```

GPT-4 Model (Model 2)

- Accuracy: 94.6%
- Validation Accuracy: 93.5%





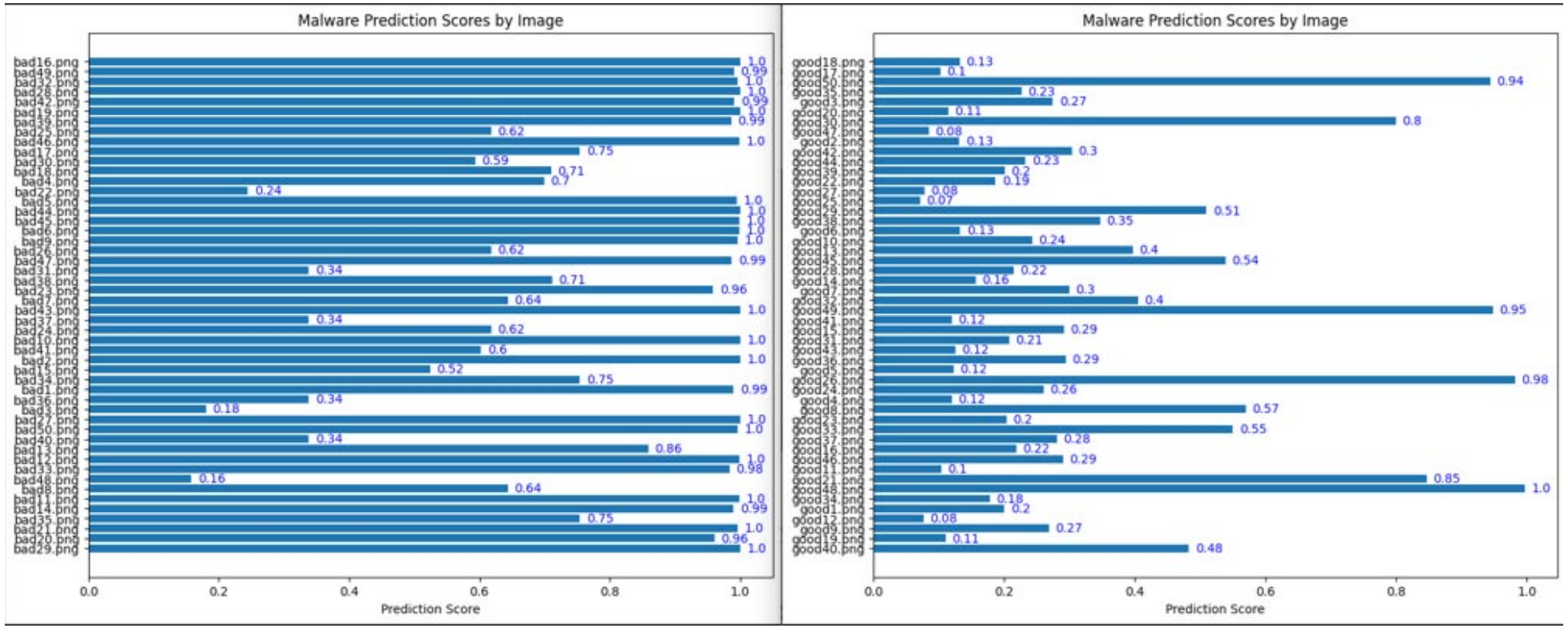
UNG | UNIVERSITY of NORTH GEORGIA™

Comparison: Windows Defender

- Total Accuracy: 96%

File Type	Total Files	Flagged
Benign	50	0
Malicious	50	46

Comparison: Model 1

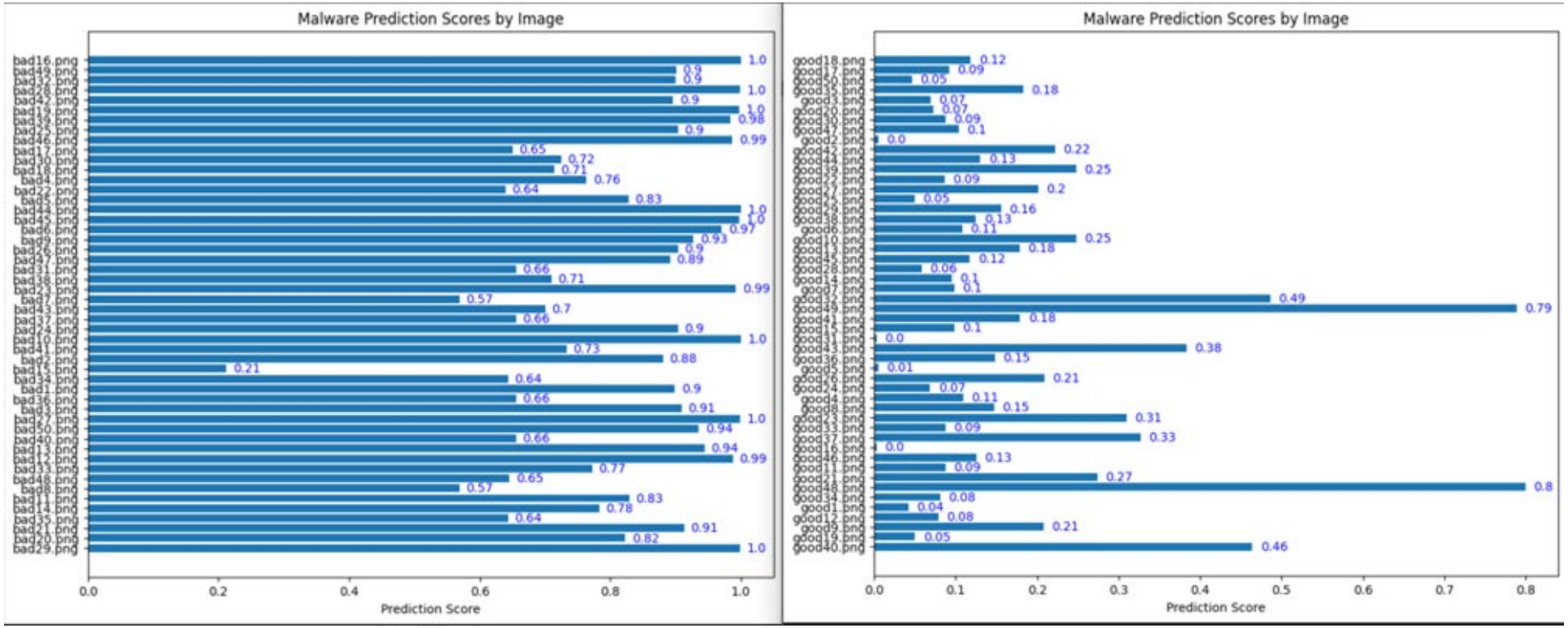


- Total Accuracy: 83%

File Type	Total Files	Flagged
Benign	50	10
Malicious	50	43

UNG | UNIVERSITY of NORTH GEORGIA™

Comparison: Model 2 (GPT)



- Total Accuracy: 97%

File Type	Total Files	Flagged
Benign	50	2
Malicious	50	49

UNG | UNIVERSITY of NORTH GEORGIA™

Final Results

	Model 1	Model 2	Windows Defender
Benign %	80%	96%	100%
Malicious %	86%	98%	92%
Total Acc %	83%	97%	96%

UNG | UNIVERSITY of NORTH GEORGIA™

Key Takeaways

- Convolutional Neural Networks (CNNs) are highly effective malware detection models, as they identify unique patterns of malware in grayscale images.
- They offer comparable or superior accuracy to industry standard methods.
- The accuracy, combined with the advantages of zero-day malware detection, makes them viable for antivirus applications.

THANK YOU FOR YOUR ATTENTION

Q/A

- Dr. Hyungbae Park, GCFE, GREM, GCLD, GMLE
- Email: hpark@ung.edu

UNG | UNIVERSITY of NORTH GEORGIA™