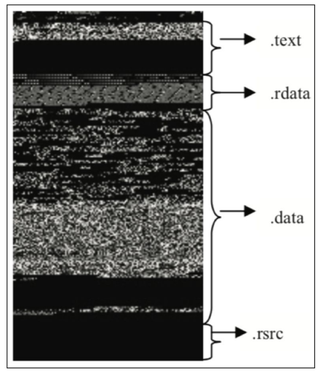
An analysis of different xAI methods for explaining malware image classifications

Bram de Jonge
Technische Universiteit Delft

Introduction

- Malware analysis is usually done through either static or dynamic analysis
- Another malware analysis method was introduced in 2011, using grayscale byteplot images based on malware binaries
- CNNs trained on malware byteplot images achieve very high classification accuracies, but we do not know why
- Limited Explainable AI research has been done on these models, and they do not link back their findings to the actual file sections of malware samples
- Goal is to find out what a CNN actually looks at, and which xAI technique showcases this the best

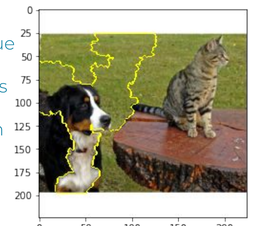


Explainable AI

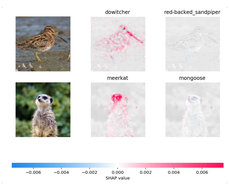
There are many xAI techniques in the field. The ones used in this research are:

- Grad-CAM
  Gradient based method that highlights regions of the input image that contribute most to a prediction. Highlighted regions can be mapped back to code in samples to find important sections/strings by heatmap weight.

- LIME
  Model agnostic technique that aims to reconstruct local decision boundaries to obtain a more interpretable model. Can also be used to show regions working against a certain prediction.



- SHAP
  Explainability framework based on Shapley values from cooperative game theory. Combinations of different image regions instead of pixels are tested to see how much each image region contributes to each output



Research Question

Which Explainable AI method is most effective for explaining CNN decisions on malware byteplot images, and how do the methods differ in their explanations?

Subquestions:

1. Which file sections receive the most attention, according to xAI techniques?
2. When occluding high attention sections found by xAI, does accuracy indeed drop?
3. How do the different xAI techniques compare in terms of effectiveness?
4. What are the differences between the sections highlighted by different xAI methods?

Methodology

- Dataset containing 10,010 malware samples provided by supervisor, both unpacked and packed using different methods. Dataset contains both original binaries and corresponding Nataraj images.
- Nataraj images are resized to 244 x 244 pixels to match the input required for ResNet-18. They are also normalized using means and standard deviations from the ImageNet dataset
- Pipeline consists of 4 stages:
  - Baseline classifier is trained from a ResNet-18 base. Training is done on an RTX 4070 across 30 epochs, using a 70-15-15 dataset split. Test set is saved for use in subsequent experiments
  - All three xAI methods are applied to the test set, with their results being saved. Grad-CAM is applied to the final layer, LIME is done with 500 samples and SHAP is done with 100
  - Attention scores in the image are mapped back to the original file sections in the binaries. Since the images get resized to fit the ResNet-18 structure, a one-to-one mapping of pixels to bytes is not possible. A row level approximation is used instead
  - To verify the importance of highlighted sections, they are occluded to check if accuracy does experience a great drop. This is done by zero-ing out the chosen sections. This also functions as a way to compare the effectiveness of the xAI methods.

Results

SQ1: Which file sections receive the most attention, according to xAI techniques?

For PE binaries, all three methods rank the same sections in their top 3: .rdata, .eh_frame and .data.

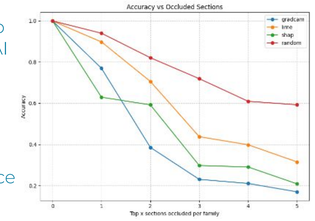
Section	Grad-CAM	LIME	SHAP
.rdata	**0.692**	**0.619**	**0.420**
.eh_frame	0.685	0.357	0.389
.data	0.634	0.532	0.410

For ELF binaries, attention is spread out more, with SHAP notably deviating quite a lot from the other methods

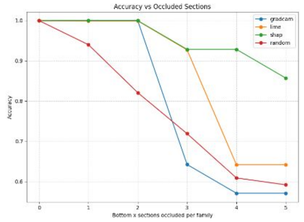
Section	Grad-CAM	LIME	SHAP
.fini	**0.572**	0.330	0.237
.rodata	0.548	**0.340**	0.251
.fini_array	0.535	0.298	0.159

SQ2: When occluding high attention sections found by xAI, does accuracy indeed drop?

When occluding the top sections identified by xAI methods, all three xAI methods perform better than a random baseline. SHAP and Grad-CAM have comparable performance with early steep drops, while LIME lags behind



For the opposite experiment of occluding the bottom sections, Grad-CAM and LIME both drop below the random baseline, while SHAP maintains a steady accuracy

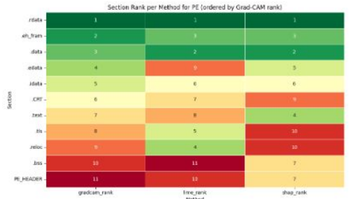


SQ3: How do the different xAI techniques compare in terms of effectiveness?

Most of this comes from the above results. For efficiency, Grad-CAM clearly performs better than the rest, taking only 35s on the test set. LIME and SHAP have comparable runtimes, taking 31m15s and 27m23s respectively

SQ4: What are the differences between the sections highlighted by different xAI methods?

Spearman rank correlation shows all three methods broadly agree on which sections are most important ($\rho > 0.90$). Despite this, there are still some clear differences that can be found between methods.



Discussion

- .rdata and .data being ranked highly makes sense. Both regions store initialized data, with .rdata storing especially important data such as read-only strings.
- .eh_frame being important is surprising. This sections contains exception handling data and is almost always mechanically generated by the compiler. This shows that the classifier may be using the tools used by the malware creator to narrow down the possible families.
- Occlusion experiments show that the xAI methods are able to identify genuinely important sections. For identifying unimportant sections however, only SHAP seems reliable
- Some limitations of this research:
  - Dataset consists of neutered samples
  - Only one configuration was used for each xAI method
  - Section mapping cannot be done one-to-one
  - Findings are specific to ResNet-18 and the dataset

Conclusion

- All three methods consistently identify .rdata, .data, and .eh_frame as the most important sections for PE binary classification, with .rdata ranked highest across all methods. For ELF binaries, attention is distributed more broadly across sections
- The occlusion experiment confirms that all three xAI methods identify genuinely important sections, and SHAP is also able to identify unimportant sections
- Combining the findings shows that classifiers primarily focus on local features within sections, as opposed to global image features
- The .eh_frame finding is a major concern, as an adversary could easily modify or strip this section. Classifiers should be thoroughly analyzed for issues like this before ever being used in production
- Future work should examine whether these findings generalize to other architectures and datasets. Applying these methods to a classifier that does not require resizing should also greatly increase the accuracy of findings.

See also:

The paper for this research can be found at:
https://resolver.tudelft.nl/uuid:d283eb65-6711-4ec8-9fa0-5e564d217b99

The code for this research can be found at:
https://github.com/BennbuildVGC/xai-malimg