Activation Analysis of a Byte-based Deep Neural Network for Malware Classification

Scott Coull
Sr. Manager, Data Science

Christopher Gardner
Reverse Engineer

Byte-based Malware Classifiers

- Feature engineering for malware classification tasks is **hard**. Can deep learning do it for us?
- Convolutional neural networks (CNNs) **automatically and efficiently learn feature representations** directly from data
- Recent work has shown **promising results** competitive with (though not better than) traditional machine learning
  - Accuracy: 90-96%, AUC: 0.96-0.98


FireEye
SECURITY REIMAGINED

Representation Learning for Malware Classification

Jeffrey Johns

Workshop track - ICLR 2018

DEEP CONVOLUTIONAL MALWARE CLASSIFIERS CAN LEARN FROM RAW EXECUTABLES AND LABELS ONLY

Marek Krčál, Czech Academy of Sciences, krcal@cs.cas.cz
Ondřej Švec, Otakar Jašek, Avast Interns, {ond.svec,jasek.ota}@gmail.com
Martin Bálek, Avast, balek@avast.com

ABSTRACT

Malware Detection by Eating a Whole EXE

Edward Raff[1,3,4], Jon Barker[2], Jared Sylvester[1,3], Robert Brandon[1,3,4]
Bryan Catanzaro[2], Charles Nicholas[4]
[1]Laboratory for Physical Sciences, [2]NVIDIA, [3]Booz Allen Hamilton, [4]University of Maryland, Baltimore County
{edraff,jared,rbrandon}@lps.umd.edu, {jbarker,bcatanzaro}@nvidia.com, nicholas@umbc.edu

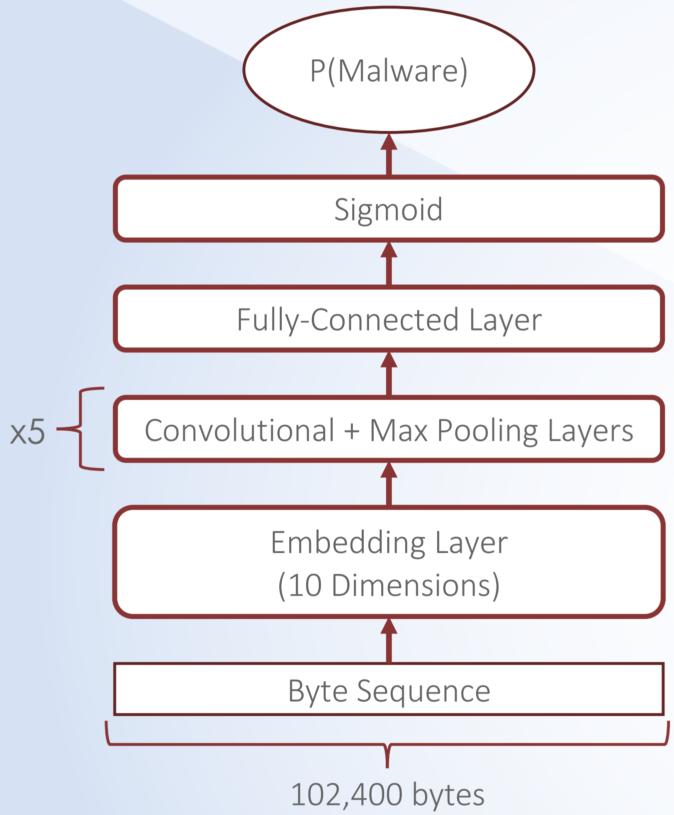
25 Oct 2017

Abstract

In this work we introduce malware detection from raw byte sequences as a fruitful research area to the larger machine learning community. Building a neural network for such a problem presents a number of interesting challenges that have not occurred in tasks such as image processing or NLP. In particular, we note that detection from raw bytes presents a sequence problem with over two million time steps and a problem where

inside a specially instrumented environment, such as a customized Virtual Machine (VM), which introduces high computational requirements. Furthermore, in some cases it is possible for malware to detect when it is being analyzed. When the malware detects an attempt to analyze it, the malware can alter its behavior, allowing it to avoid discovery (Raffetseder, Kruegel, and Kirda 2007; Garfinkel et al. 2007; Carpenter, Liston, and Skoudis 2007). Even when malware

CNN Models



- **Baseline**
  - 15.6M Windows PEs (80% goodware)
  - July 2015 to July 2017
  - Stratified sampling
- **Small**
  - 7.3M Windows PEs (50% goodware)
  - July 2016 to November 2016
  - No sampling
- **Baseline+Dropout**
  - Same data as Baseline
  - Dropout layers before convolutional layers

©2019 FireEye

Model Evaluation

Model	Train Data Size	Train Data Mal:Good	Test Results F1	Test Results AUC
Small	7.27M	50:50	0.943	0.98
Baseline	15.62M	20:80	0.919	0.96
Baseline+Dropout	15.62M	20:80	0.869	0.87

16.55M binaries (50:50) from June 1, 2018 to August 31, 2018

Model trained on **small dataset performs noticeably better** despite older data and fewer samples

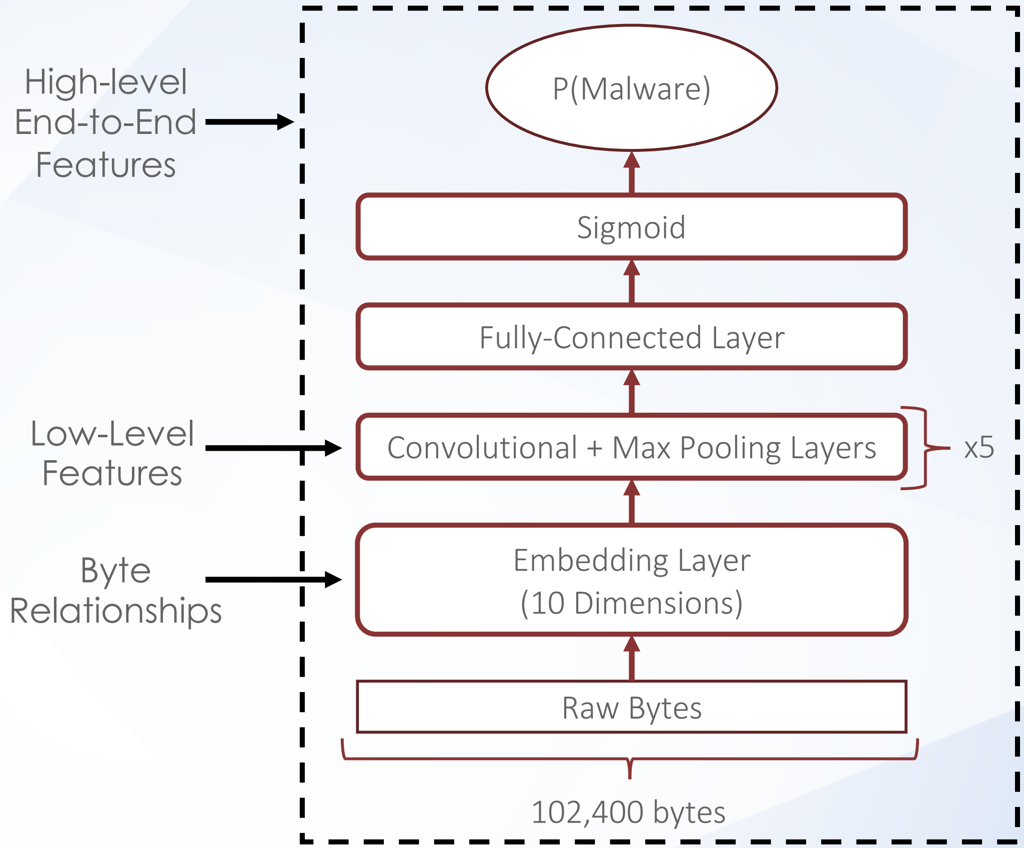
©2019 FireEye

What are byte-based malware classifiers learning?

What is the impact of dataset volume and regularization on learned features?

Analysis Overview

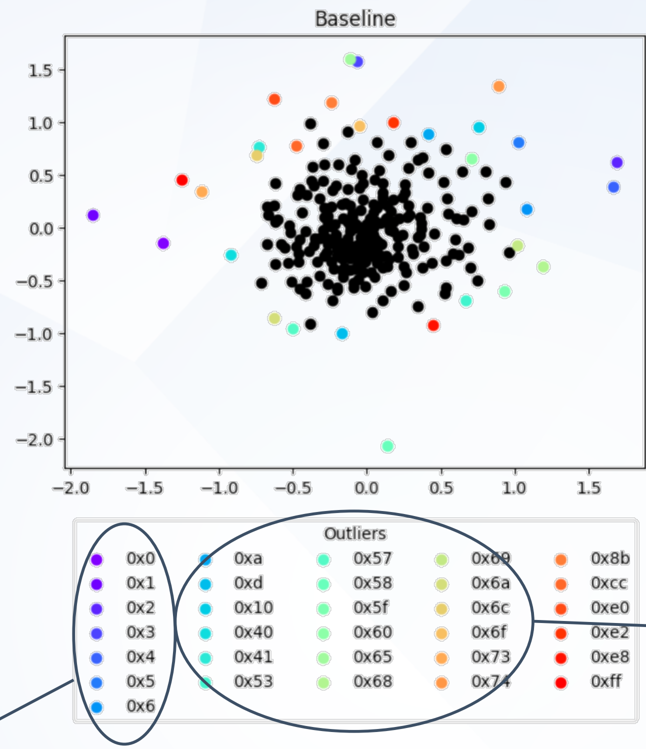
- **Cluster** and **visualize** embedding layer with HDBSCAN[1] and MDS[2]
- **Disassembly of byte sequences** with large activations in first convolutional layer
- **End-to-end analysis** of byte segments with GradientSHAP[3]



[1] Campello, Ricardo JGB, Davoud Moulavi, and Jörg Sander. *Density-based clustering based on hierarchical density estimates.*
[2] I. Borg and P. Groenen, *Modern Multidimensional Scaling. Theory and Applications*.
[3] Lundberg, Scott M., and Su-In Lee. *A unified approach to interpreting model predictions.*

©2019 FireEye

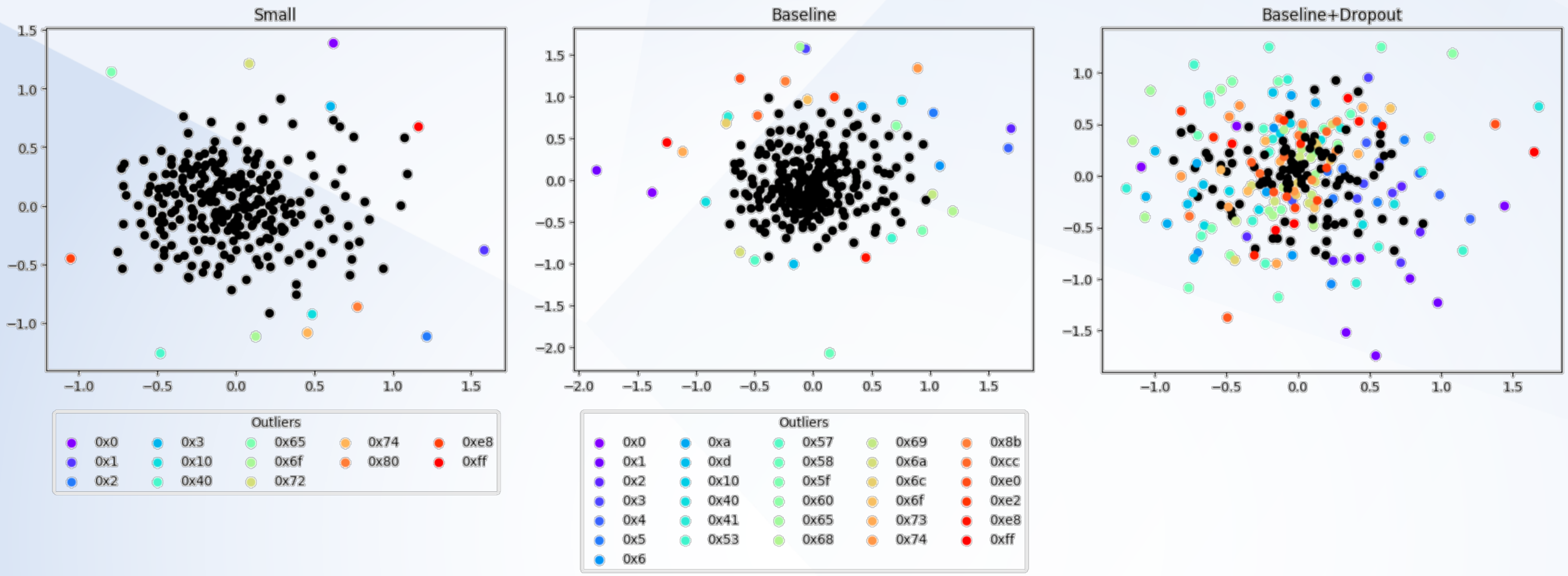
Byte Embeddings



eax-edx
padding
short jumps

ASCII Characters
@, \n, A,..., j, s, t

Byte Embeddings

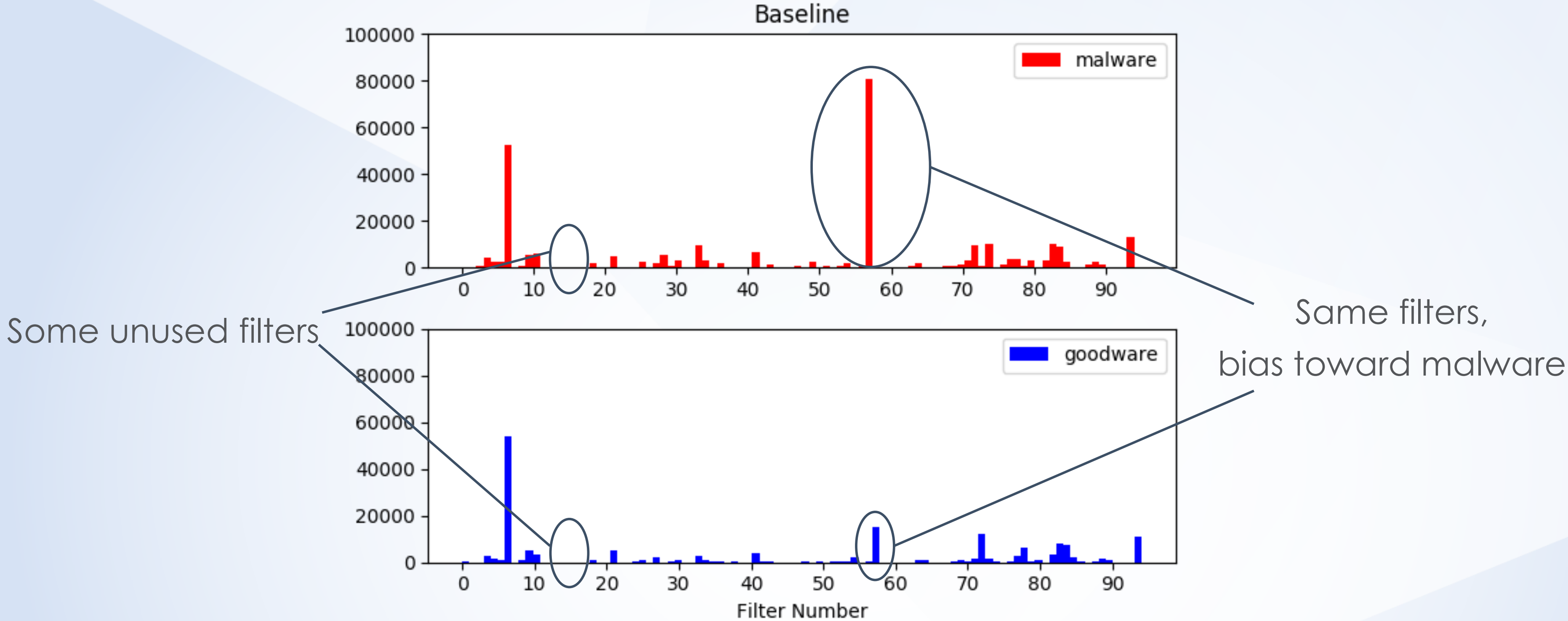


Increase in number of outliers with more data/regularization

Learned features appear to be less flexible

Low-Level Feature Detectors

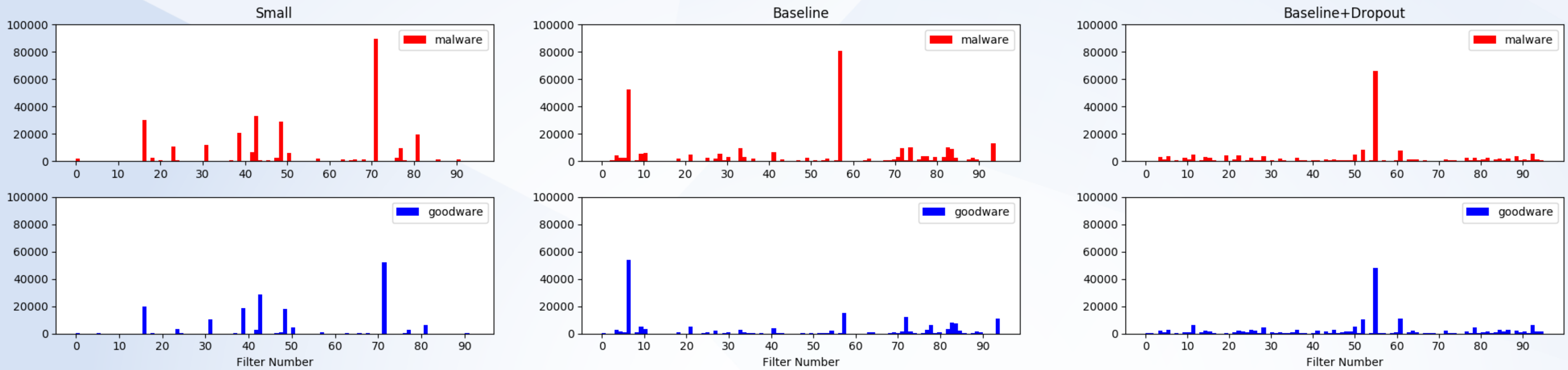
Distribution of Top-100 Activations Across First-Level Filters



©2019 FireEye

Low-Level Feature Detectors

Distribution of Top-100 Activations Across First-Level Filters



More data and regularization appears to lead to more features that are equally applicable across the two classes

Supports earlier observation about feature specificity

©2019 FireEye

Low-Level Feature Detectors

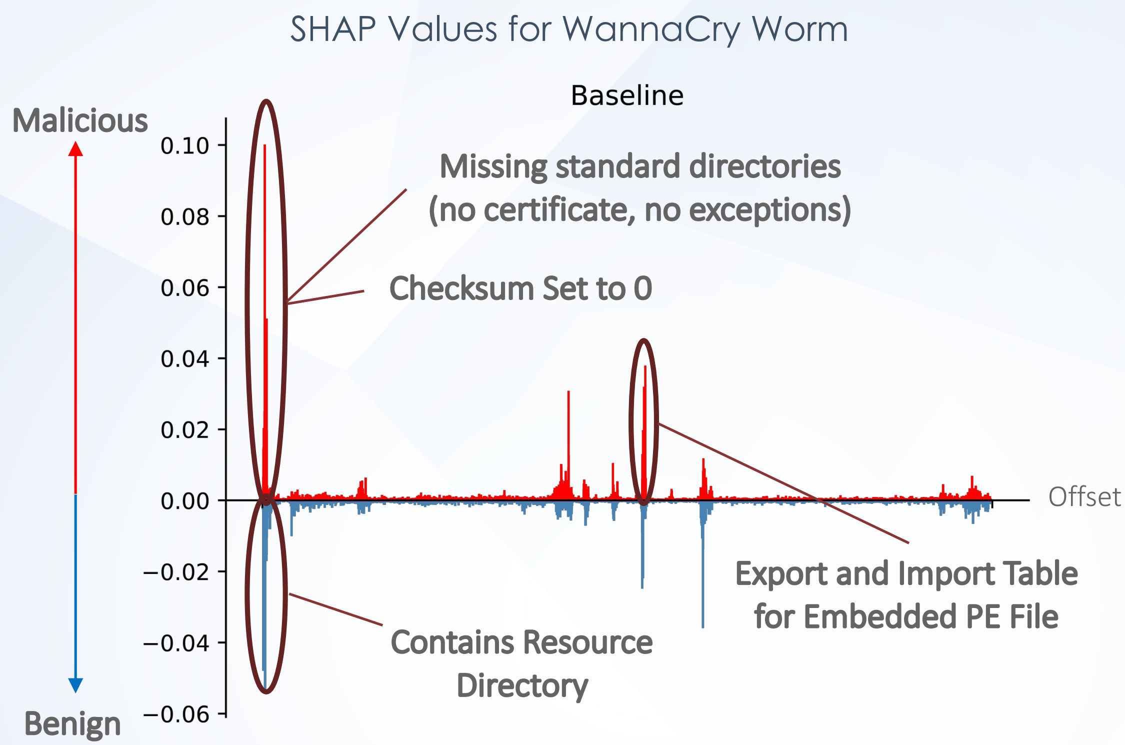
Loose filters

Specific filters

Model	Features: Strings	Features: Instructions
Small	**Filter 71: 'C', 'r', '@'**	**Filter 16: Push sequences**
	(0x40f0c8L): tGenKey.	(0x10007edbL): je,0x10007ff1
	(0x40f0d0L): CryptDec	(0x10007ee1L): push,0xff
	(0x40f0d8L): rypt....	(0x10007ee6L): push,edi
	(0x40f0e0L): CryptEnc	(0x10007ee7L): push,0x10007ca5
	(0x40f0e8L): rypt....	(0x10007eecL): push,0x4
Baseline	**Filter 83: 'r', 's'**	**Filter 57: Function calls**
	(0x40d850L): ....GetP	(0x4046b4L): push,0x0
	(0x40d858L): rocAddre	(0x4046b6L): push,0x0
	(0x40d860L): ss..R.Lo	(0x4046b8L): push,0x1
	(0x40d868L): adLibrar	(0x4046baL): push,0x0
	(0x40d870L): yA....Gl	(0x4046bcL): call,dword,15042
Baseline+Dropout	**Filter 11: 'Directory'**	**Filter 61: mov sequences**
	(0x40d9e0L): ctoryW..	(0x408d65L): je, 0x408d6a
	(0x40d9e8L): N.Create	(0x408d67L): mov, dword , edx
	(0x40d9f0L): ,Director	(0x408d6aL): mov, esi, dword
	(0x40d9f8L): yW....Ge	(0x408d6dL): mov, dword, esi
	(0x40da00L): tTempPat	(0x408d70L): mov, ecx, dword

©2019 FireEye

End-to-End Features

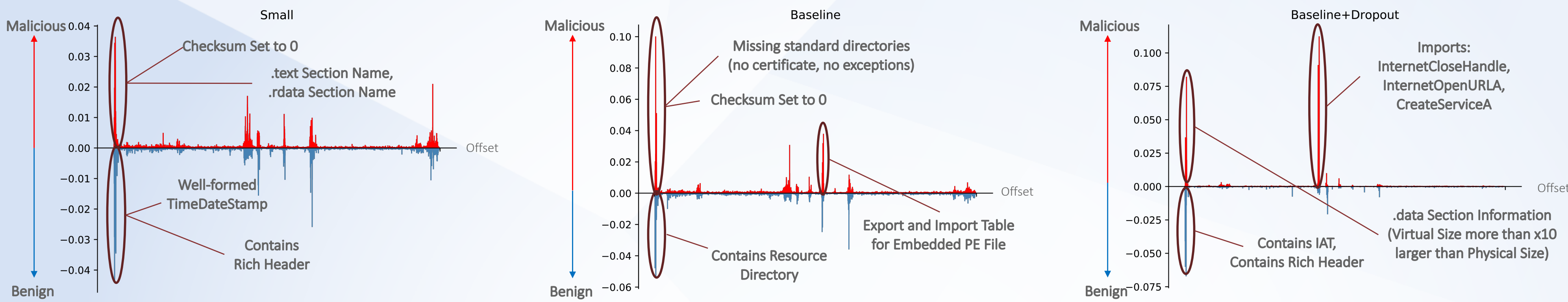


End-to-end features map closely to manual feature engineering

©2019 FireEye

End-to-End Features

SHAP Values for WannaCry Worm

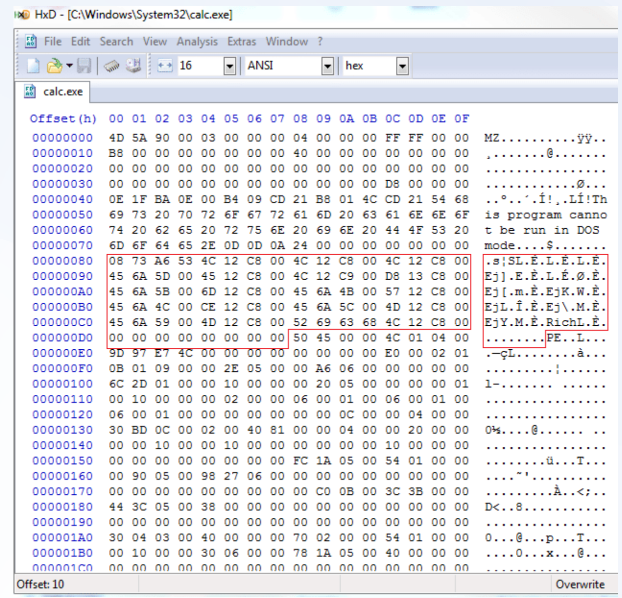


Data and regularization result in more focused areas of interest
Model appears to learn presence/absence of structural features

©2019 FireEye

The Case of the Rich Header

- Rich header is added by Microsoft's linker and contains metadata about the binary
- Should be effectively 'random' due to XOR encryption using key derived from checksum
- <u>Hypothesis</u>: Hierarchical pooling can detect presence of fixed bytes around header (e.g., 'Rich')
- Proxy for whether **non-Microsoft compiler** was used, which is **common in malware**



Summary

- CNN architectures can learn **meaningful features**
  - Imports, presence of Rich header, incorrect checksums, etc.
  - Many features mimic manually-derived features from traditional ML models
  - Partly contradicts findings by Demetrio et al. on MalConv[4]
- **Model depth**, **dataset**, and **hierarchical pooling** appear to be key
- Malware classification performance relies on **detecting malware indicators**
  - Increased data and regularization lead to more specific features that were equally applicable across the two classes but worse detection performance

©2019 FireEye

[4] L.Demetrio,B.Biggio,G.Lagorio,F.Roli,and A.Armando. *Explaining Vulnerabilities of Deep Learning to Adversarial Malware Binaries*.

FireEye Data Science is Hiring!

- Data scientist positions open at the **Senior, Staff, and Principal level**
- Perform cutting-edge ML research and apply it to cybersecurity problems
- Work on problems from across the entire cybersecurity spectrum!
  - Threat Intelligence, Email, Network, Endpoint …

Thank you!
scott.coull@fireeye.com
@DrScottCoull
https://scottcoull.com

©2019 FireEye