SAND2018-13752C

Dynamic Analysis of Executables to Detect and Characterized Malware

PRESENTED BY

Michael R. Smith

Story to Tell

Malware and cyber attacks are occurring at an increasing rate
- Heartbleed
- Ransomware
- Attacks on power systems

A common thread is that something has to be executed on the host system
- Signature based malware detection susceptible to obfuscation attacks
  - Add null operators changes hashes
  - Cannot detect novel variants of executables
  - Brittle
- Current ML approaches
  - Static or Dynamic

Dynamic analysis monitors the system calls (calls from an executable to underlying OS API)

Prevalence of Malware and Cyber Attacks

New Linux malware mines cryptocurrency and steals your password

Amazon hit with major data breach days before Black Friday

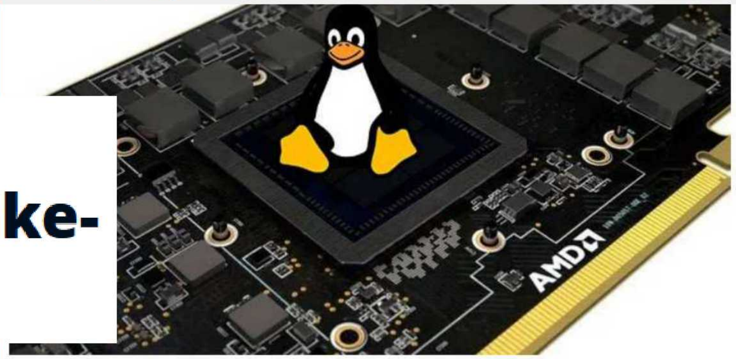
Customers' names and email addresses posted on website, tech giant confirms

Nasty New Linux Crypto Malware Compromises Root, Launches DDoS Attacks

With the value of Bitcoin once again experiencing a big drop this past week, you may begin to think that malware developers would begin shifting focus elsewhere. Unfortunately, that's far from being the case. Even if crypto seems to have modest value, that value becomes substantial when you multiply it

Ukraine detects new Pterodo backdoor malware, warns of Russian cyberattack

Revived Gamaredon threat group just part of wave of new attacks tied to Russia's FSB.



Cybersecurity Firm Detects Cryptojacking Malware on Make-A-Wish Foundation Website

Emotet malware runs on a dual infrastructure to avoid downtime and takedowns

Researchers spot unique design in the server infrastructure propping up the Emotet malware.

Google removes 13 malware apps from its Play Store

The 13 malware apps have been downloaded over 5,60,000 times from the Play Store

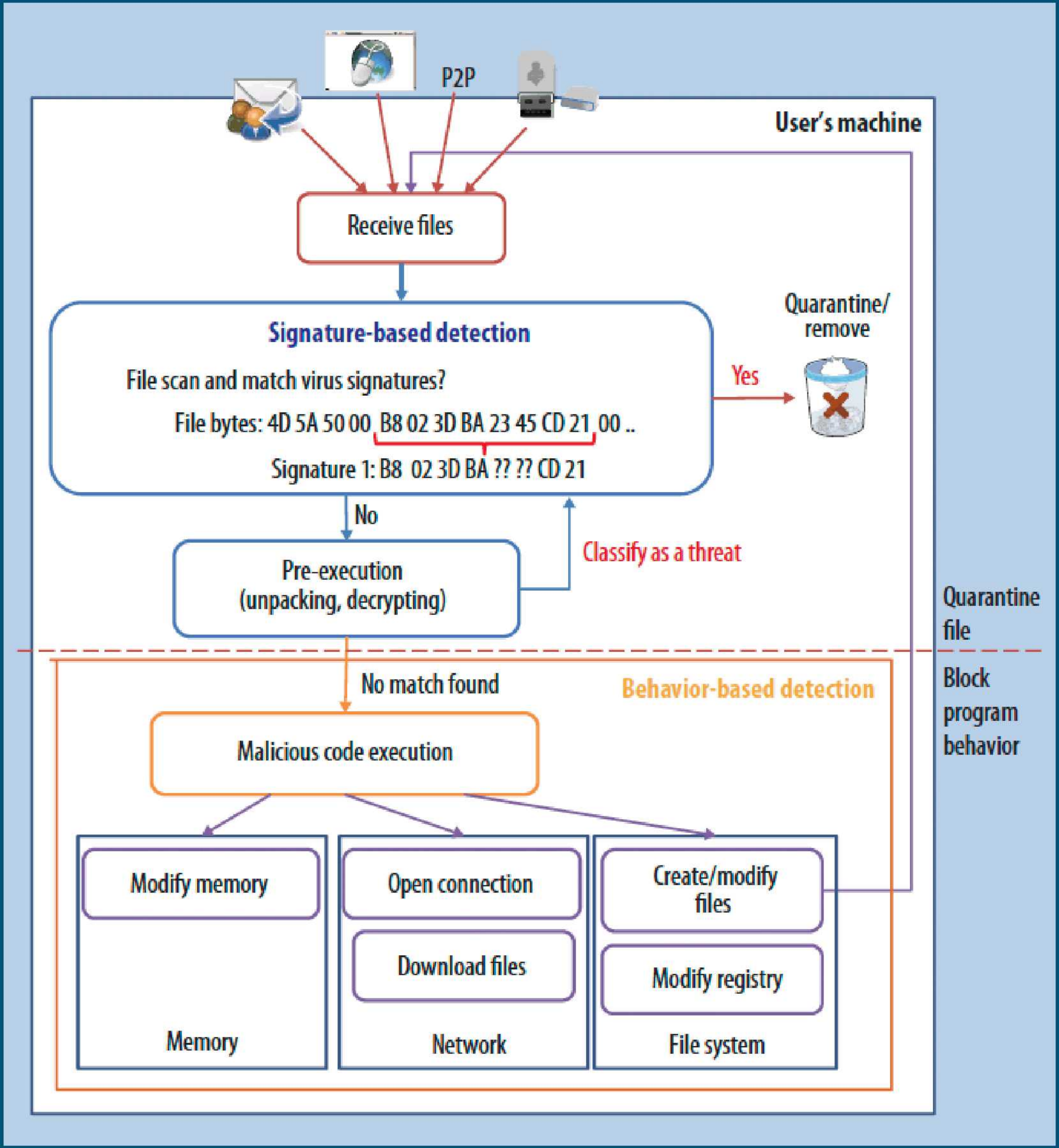
Current Antivirus Techniques

Signature Matching
- Easiest to defeat
- Trivially modify with non-used code

Heuristics-based Detection
- Look for generic characteristics
  - Specific, rare, operations
  - Specific registry modifications

Behavioral Detection
- Run the executable in a sandbox
- Observe behavior

Each is vulnerable to minor changes in the code
- Not able to adapt to new changes



J. Hoe, H. Kim and O. Sukwong, "Commercial Antivirus Software Effectiveness: An Empirical Study," in *Computer*, vol. 44, no. , pp. 63-70, 2010. doi:10.1109/MC.2010.187

Thwarting Anti-Virus

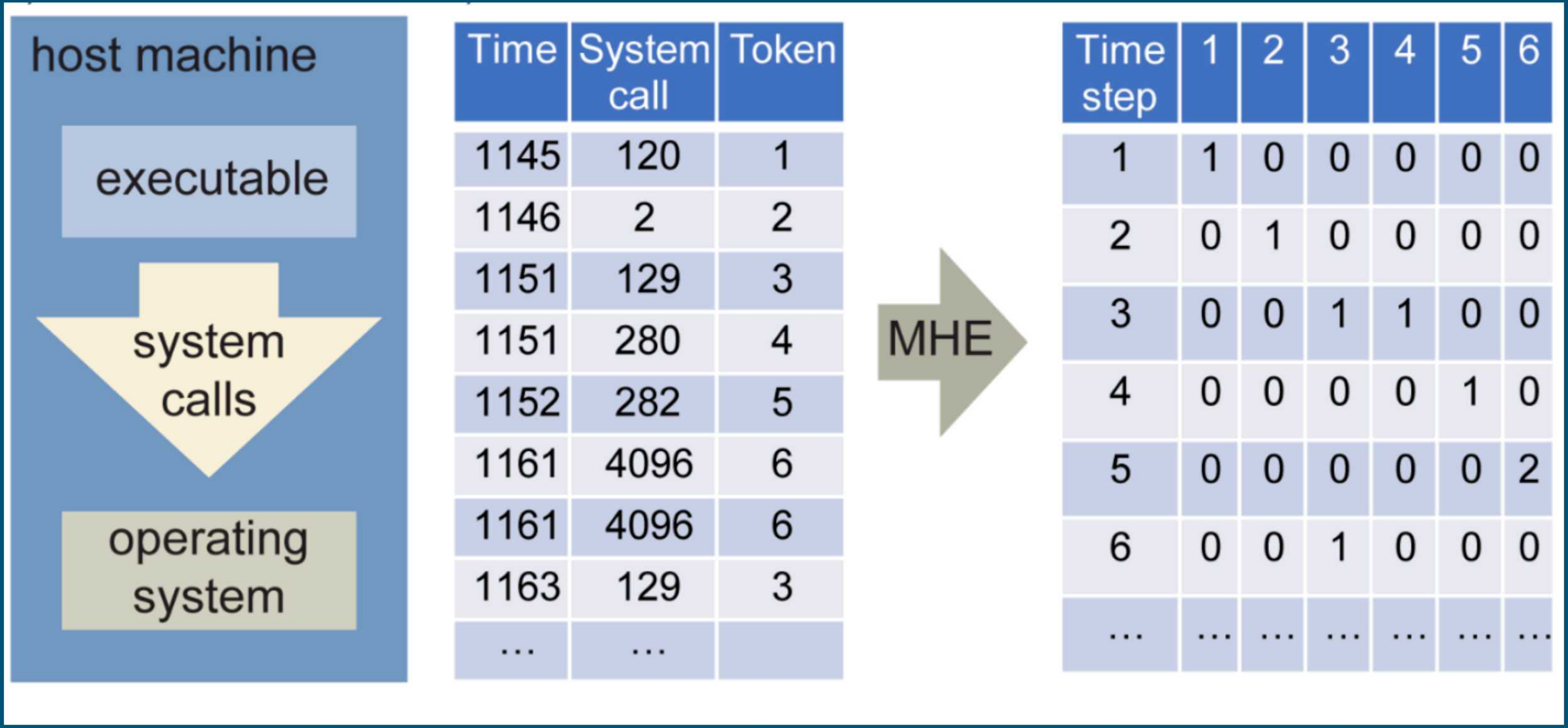
- Code packing and encryption
- Code mutation
- Polymorphic code
- Stealth techniques
  - Process injection
  - Process hiding
- Turning off anti-virus
- Adding noop

Solution: Use machine learning to monitor system calls
- **ML can generalize away from static signatures**
- **System calls are the base level for interacting with the operating system**

Collecting System Call Data

Executables collected from a corporate gateway 2012

- Assumed benign if when ran through several anti-virus engines none were alerted
- Malware was collected from Arbor Networks daily feeds
- Windows 32-bit executables
- 14,483 samples: 6,197 benign and 8,286 malicious



Time	System call	Token
1145	120	1
1146	2	2
1151	129	3
1151	280	4
1152	282	5
1161	4096	6
1161	4096	6
1163	129	3
…	…	

Time step	1	2	3	4	5	6
1	1	0	0	0	0	0
2	0	1	0	0	0	0
3	0	0	1	1	0	0
4	0	0	0	0	1	0
5	0	0	0	0	0	2
6	0	0	1	0	0	0
…	…	…	…	…	…	…

Machine Learning in Detecting Malware

Static Analysis

- Extracts features from the executable without running it
  - Statistics and meta-features
    - List of DLLs
    - Byte n-gram
- Vulnerable to obfuscation techniques

Dynamic Analysis

- Runs an executable to extract features
  - List of system calls
  - Libraries loaded
  - Writing to registries

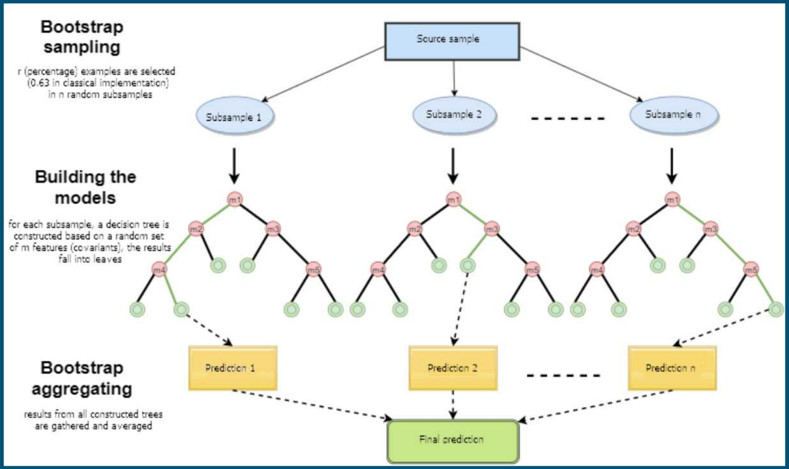
Previous work general reports 96-98% classification accuracy

- Several caveats:
  - Generally cross-fold validation (what about concept drift)
  - Class imbalance is generally not studied (a small percentage of executables are malware)

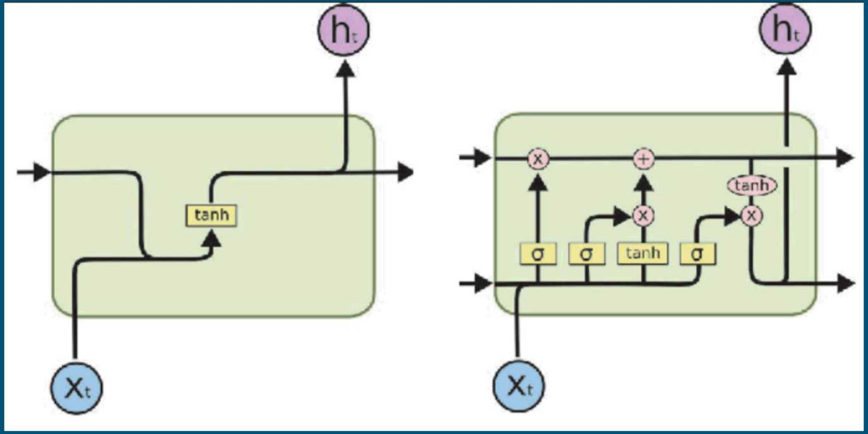
How well would deep learning methods in detecting malware using dynamic analysis?

Examined Learning Algorithms

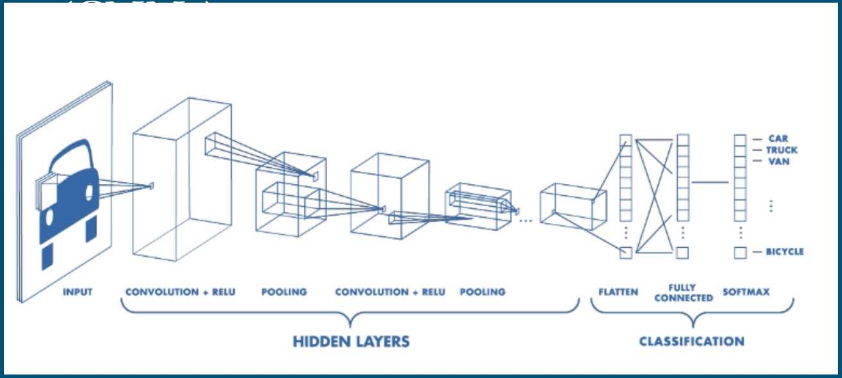
Histograms with Random Forests


Bootstrap sampling
r (percentage) examples are selected (0.63 in classical implementation) in n random subsamples
Source sample
Subsample 1
Subsample 2
Subsample n
Building the models
for each subsample, a decision tree is constructed based on a random set of m features (covariants), the results fall into leaves
m1
m2
m3
m4
m5
Bootstrap aggregating
results from all constructed trees are gathered and averaged
Prediction 1
Prediction 2
Prediction n
Final prediction

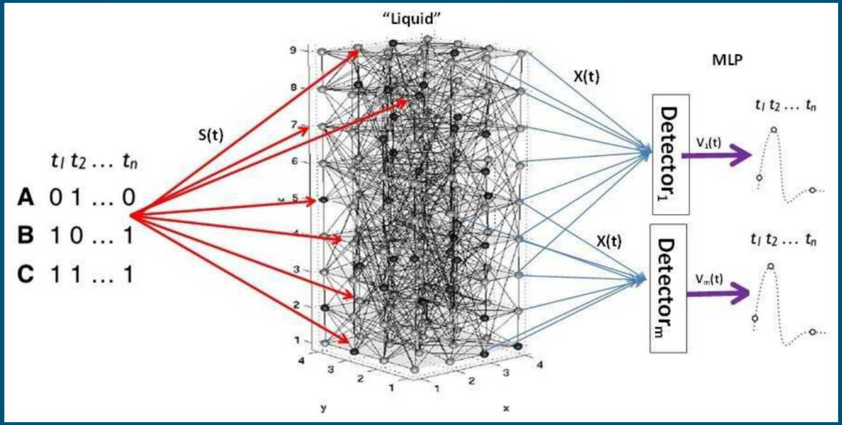
Long Short-Term Memory Recurrent Neural Networks (LSTM)



Convolutional Neural Networks



Liquid State Machines (LSMs)



Combined CNN and LSTM (CNN+LSTM)

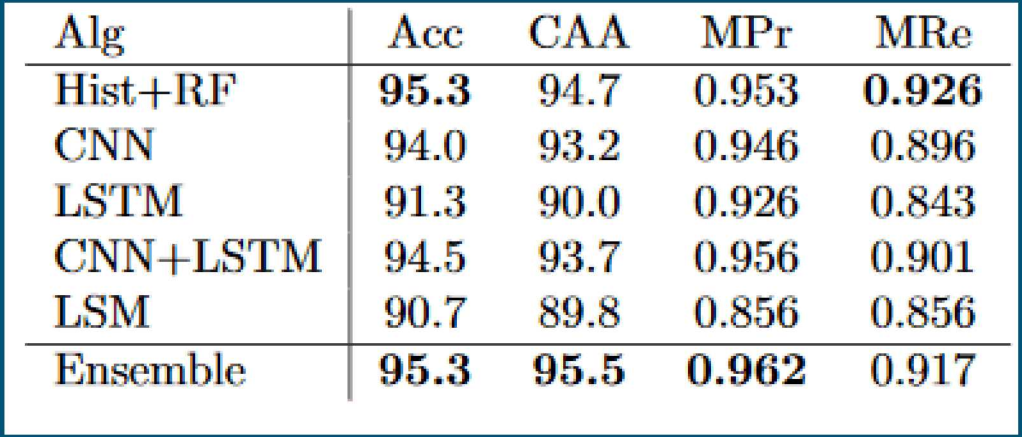
Results

1000 time steps sequence length

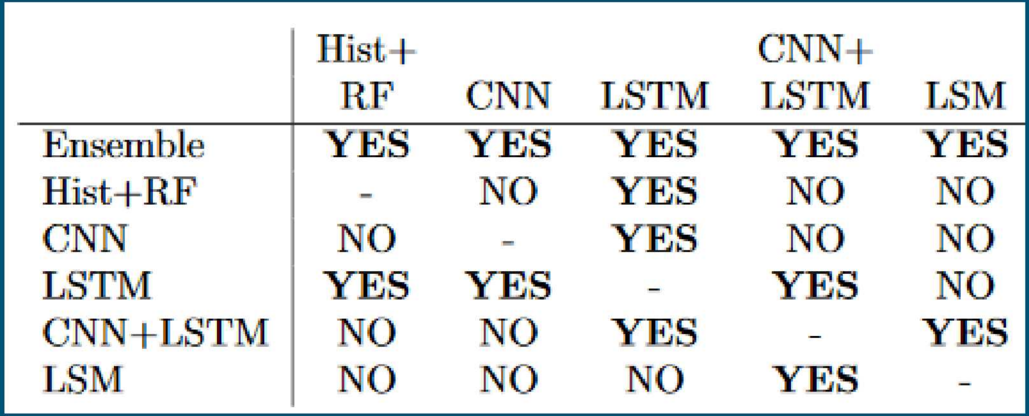
Each method achieves a class averaged accuracy greater than 90%

Random Forests out perform the deep learning approaches
- Statistical significance over the LSTM

Ensemble statistically significantly outperforms all other approaches



Alg	Acc	CAA	MPr	MRe
Hist+RF	**95.3**	94.7	0.953	**0.926**
CNN	94.0	93.2	0.946	0.896
LSTM	91.3	90.0	0.926	0.843
CNN+LSTM	94.5	93.7	0.956	0.901
LSM	90.7	89.8	0.856	0.856
Ensemble	**95.3**	**95.5**	**0.962**	0.917



	Hist+ RF	CNN	LSTM	CNN+ LSTM	LSM
Ensemble	**YES**	**YES**	**YES**	**YES**	**YES**
Hist+RF	-	NO	**YES**	NO	NO
CNN	NO	-	**YES**	NO	NO
LSTM	**YES**	**YES**	-	**YES**	NO
CNN+LSTM	NO	NO	**YES**	-	**YES**
LSM	NO	NO	NO	**YES**	-

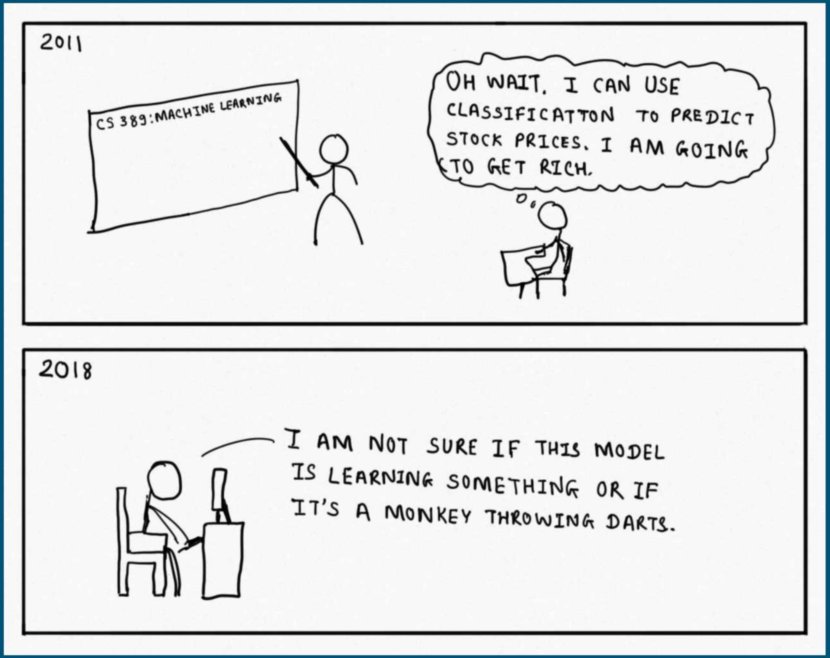
Why did deep learning not significantly out perform other methods?

A major problem in using ML in information security:

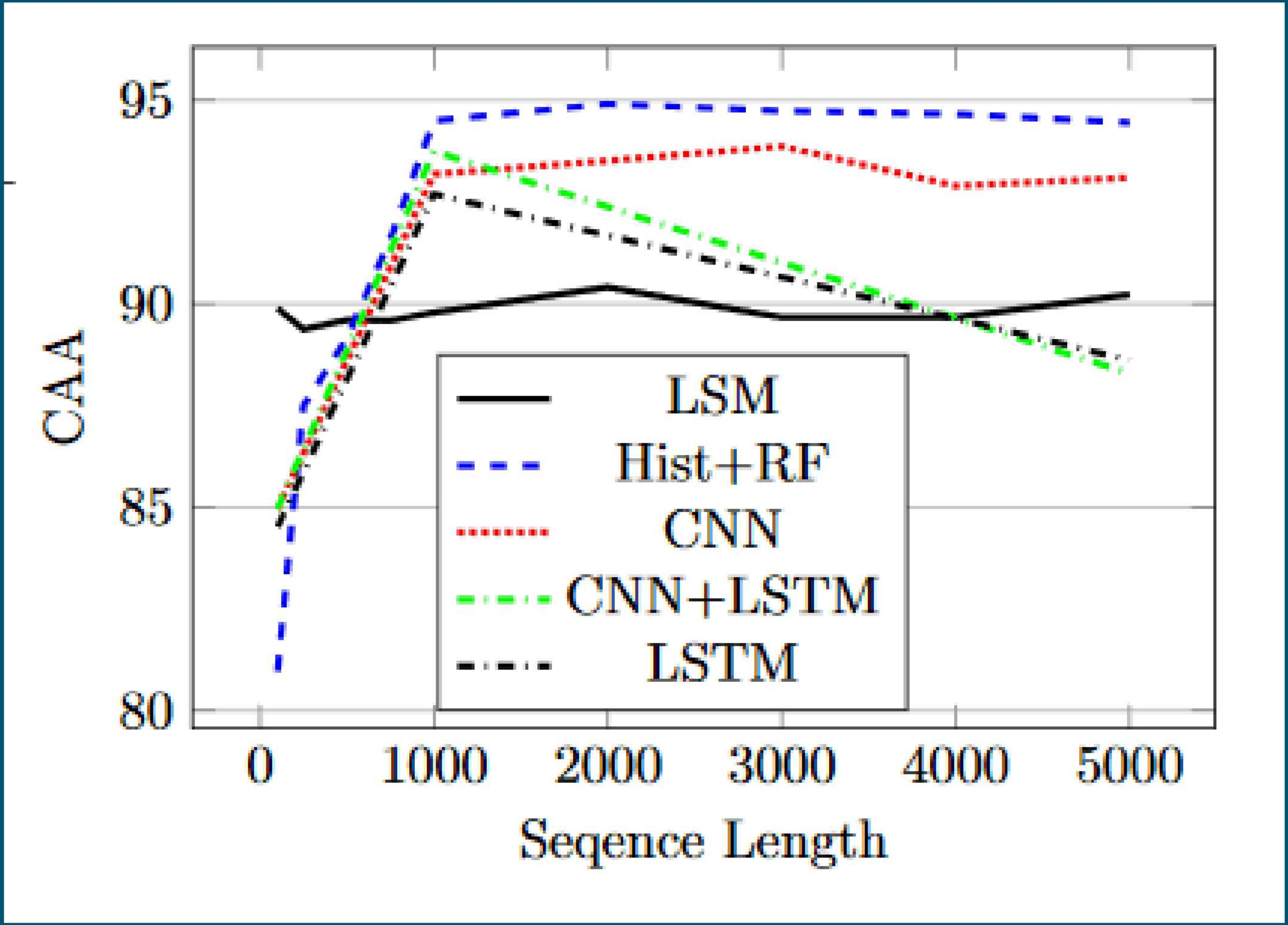
- Lack of knowledge of the domain (most users are experts in object detection)
- Space is prohibitively large and dynamic
- Environment is inherently adversarial
- Lack of labelled data

Many previous works have simply applied deep learning approached (including us)

- Need to specialize to info security domain (similar to CNNs for image processing)


2011
CS 389: MACHINE LEARNING
OH WAIT, I CAN USE CLASSIFICATION TO PREDICT STOCK PRICES. I AM GOING TO GET RICH.
2018
I AM NOT SURE IF THIS MODEL IS LEARNING SOMETHING OR IF IT'S A MONKEY THROWING DARTS.

Sequence Length



Various Learning Scenarios

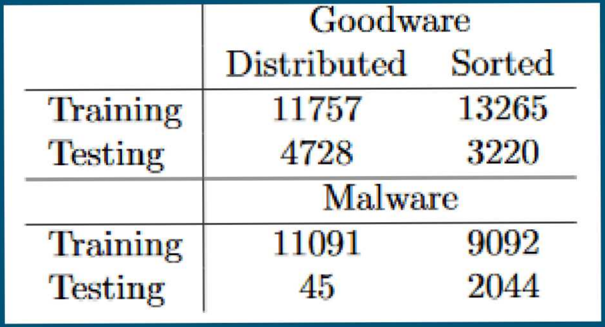
Sorted: data set with roughly balanced benign and malicious

CV: 10-fold cross validation on balanced data set

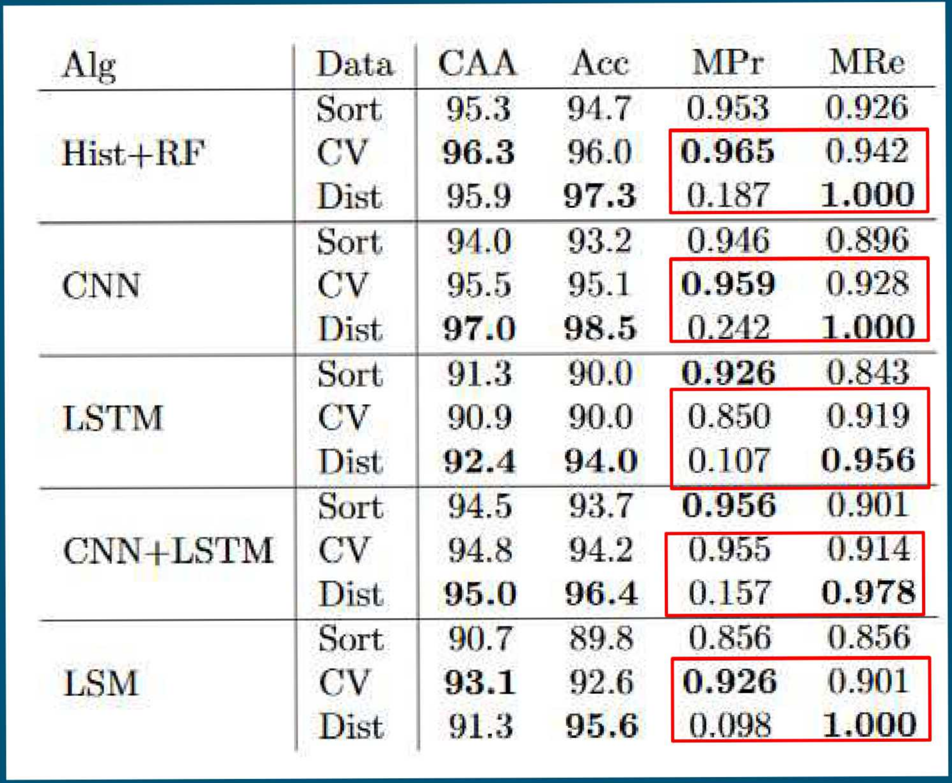
Dist: a data set with significant class skew

Key take away: CV and balanced data sets can give overly optimistic expectations

We should expect significantly lower precision



	Goodware	
	Distributed	Sorted
Training	11757	13265
Testing	4728	3220
	Malware	
Training	11091	9092
Testing	45	2044

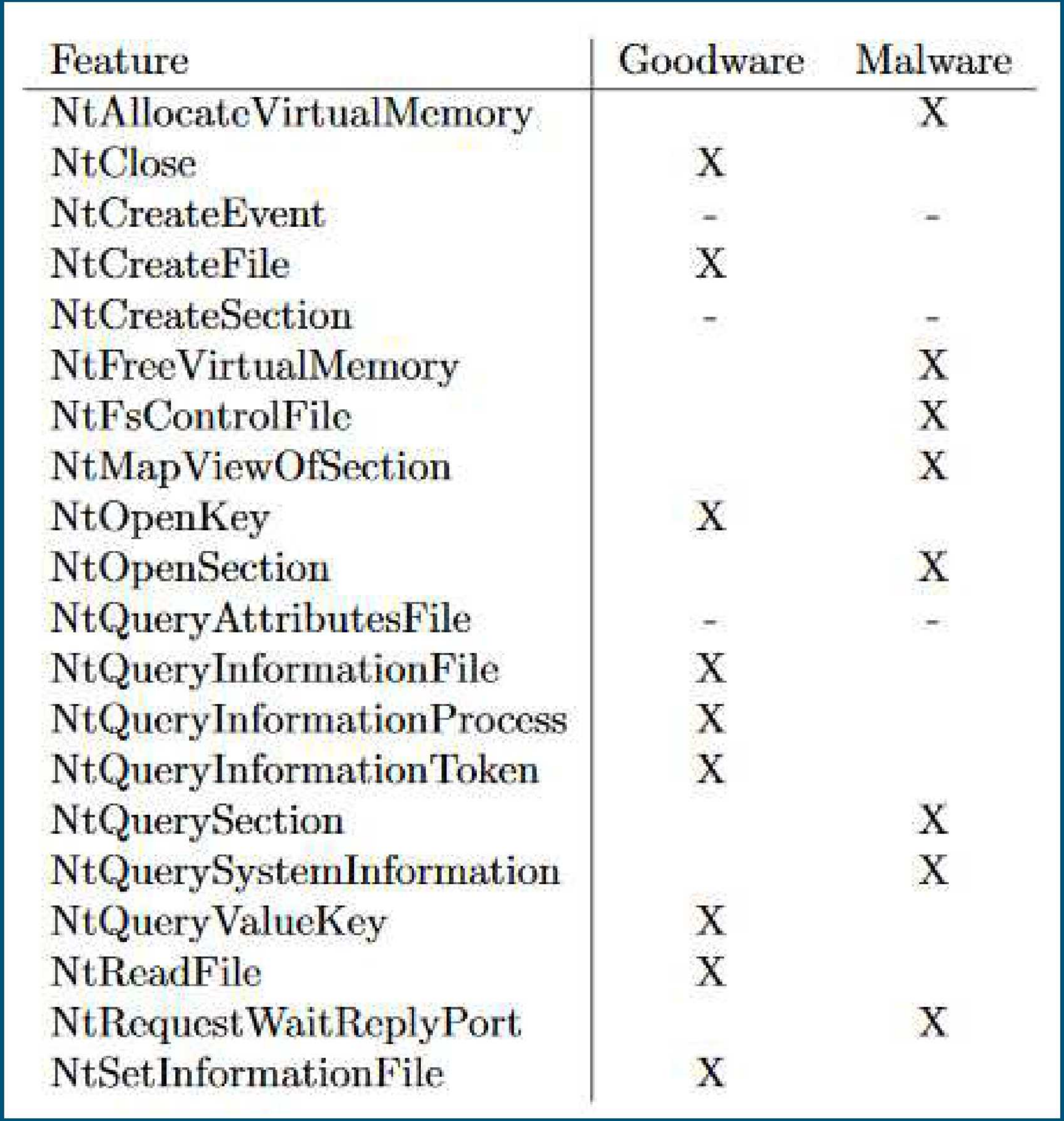


Alg	Data	CAA	Acc	MPr	MRe
Hist+RF	Sort	95.3	94.7	0.953	0.926
	CV	**96.3**	96.0	**0.965**	0.942
	Dist	95.9	**97.3**	0.187	**1.000**
CNN	Sort	94.0	93.2	0.946	0.896
	CV	95.5	95.1	**0.959**	0.928
	Dist	**97.0**	**98.5**	0.242	**1.000**
LSTM	Sort	91.3	90.0	**0.926**	0.843
	CV	90.9	90.0	0.850	0.919
	Dist	**92.4**	**94.0**	0.107	**0.956**
CNN+LSTM	Sort	94.5	93.7	**0.956**	0.901
	CV	94.8	94.2	0.955	0.914
	Dist	**95.0**	**96.4**	0.157	**0.978**
LSM	Sort	90.7	89.8	0.856	0.856
	CV	**93.1**	92.6	**0.926**	0.901
	Dist	91.3	**95.6**	0.098	**1.000**

Most Important/Discriminatory Features

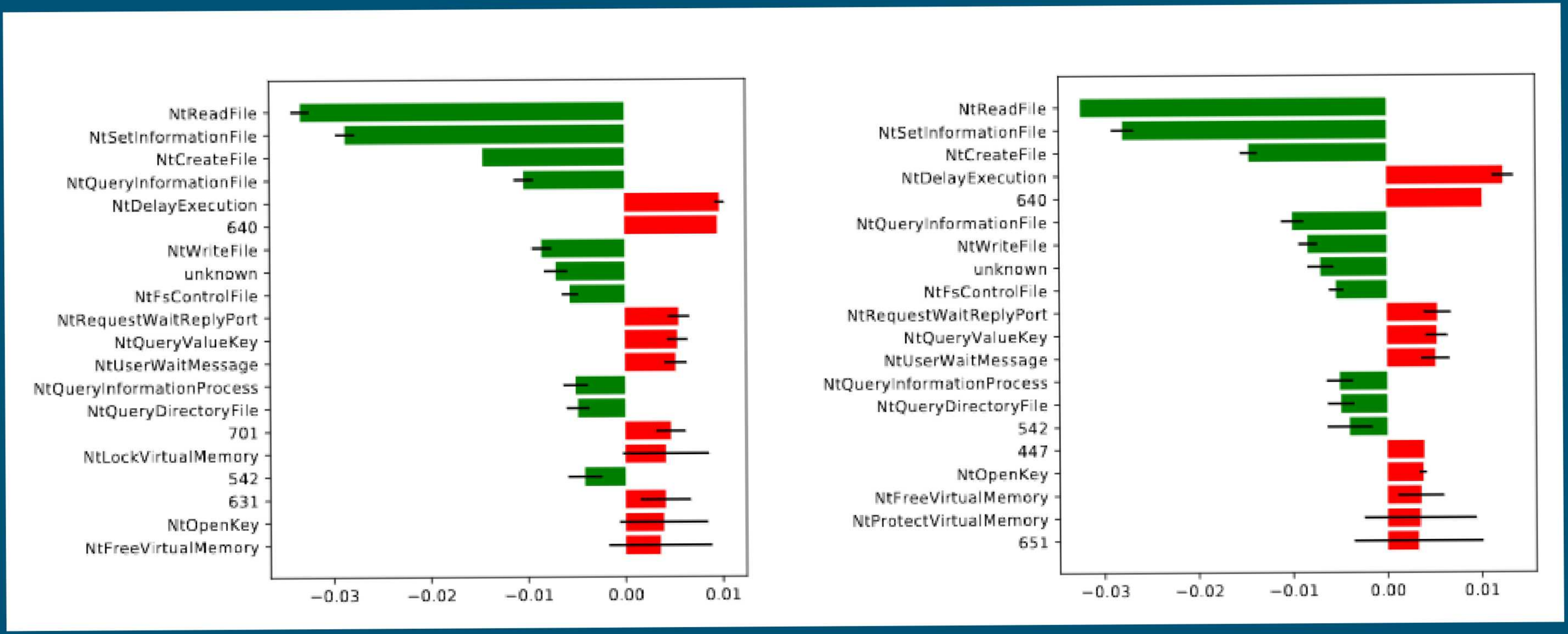
These are often used in a decision making process

- Need to be able to explain
- Build trust
- Build context



Feature	Goodware	Malware
NtAllocateVirtualMemory		X
NtClose	X	
NtCreateEvent	-	-
NtCreateFile	X	
NtCreateSection	-	-
NtFreeVirtualMemory		X
NtFsControlFile		X
NtMapViewOfSection		X
NtOpenKey	X	
NtOpenSection		X
NtQueryAttributesFile	-	-
NtQueryInformationFile	X	
NtQueryInformationProcess	X	
NtQueryInformationToken	X	
NtQuerySection		X
NtQuerySystemInformation		X
NtQueryValueKey	X	
NtReadFile	X	
NtRequestWaitReplyPort		X
NtSetInformationFile	X	

Most Important Features



Most important features for a prediction of an individual data point

- Left: Correctly classified malware samples
- Right: Misclassified malware samples
- Green: For malware
- Red: for goodware

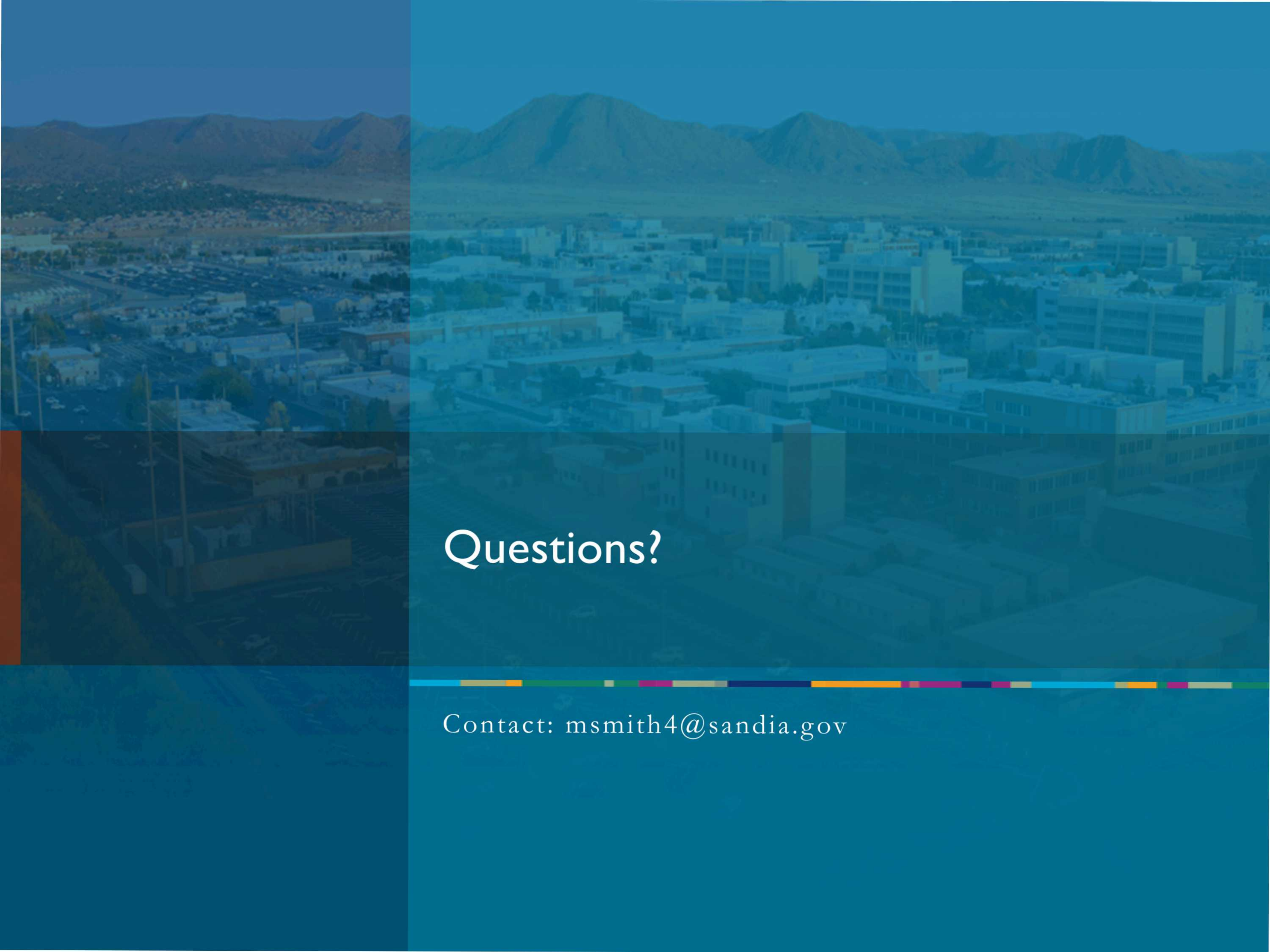
NTDelayExecution seems to flip the classification

Conclusions

Machine learning is a viable solution for malware detection
- Random Forests empirically have the best performance
- More complex algorithms are not necessarily better
- Deep learning for automatic feature extraction is still lagging in information security

Cross-validation can give overly optimistic results, especially in precision
- More realistic results come from testing on a distribution with class skew

Results need to interpreted by an analyst
- Explainability approaches are key to taking action, building trust, and improving the model
- Can help with forensics and where to put defenses



Questions?

Contact: msmith4@sandia.gov