Malware Characterization using Compiler-based Graphs

Tristan Vanderbruggen

Dept of Computer & Information Sciences

University of Delaware

Malware Characterization

Malware can:

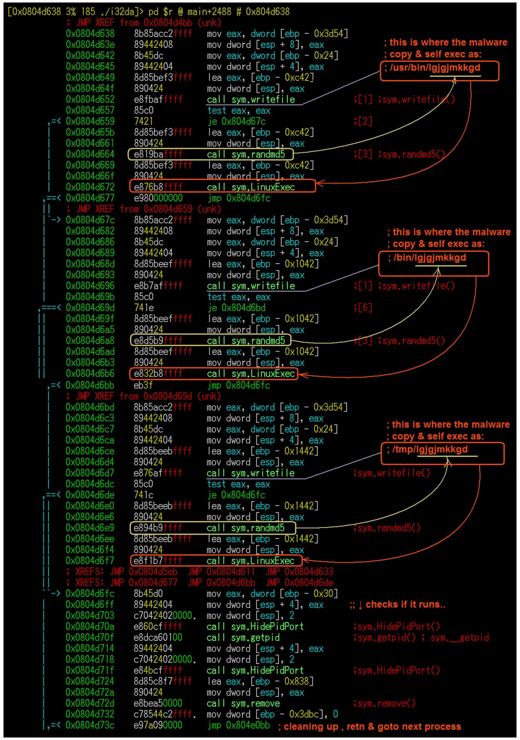
- Send spam
- Steal private information
- Ransom data
- Be used for warfare

Current technologies inadequate

- Automation => millions of variants
- Zero day exploits

Discovering new malware

- Manual analyses



Malware Characterization

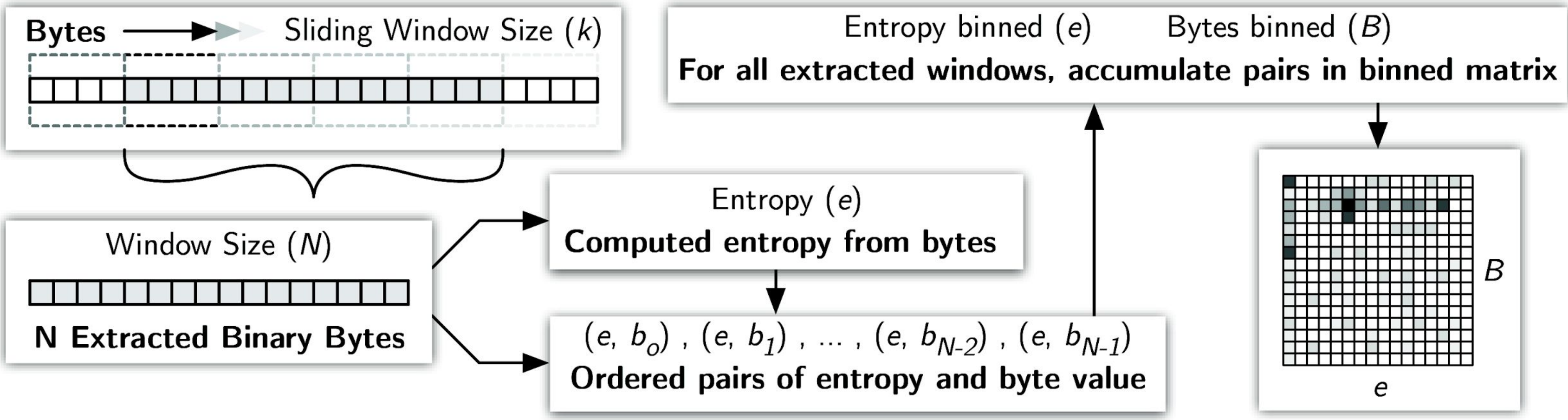
Bytes

- Shannon Entropy
- Bytes N-grams
- Strings

Codes

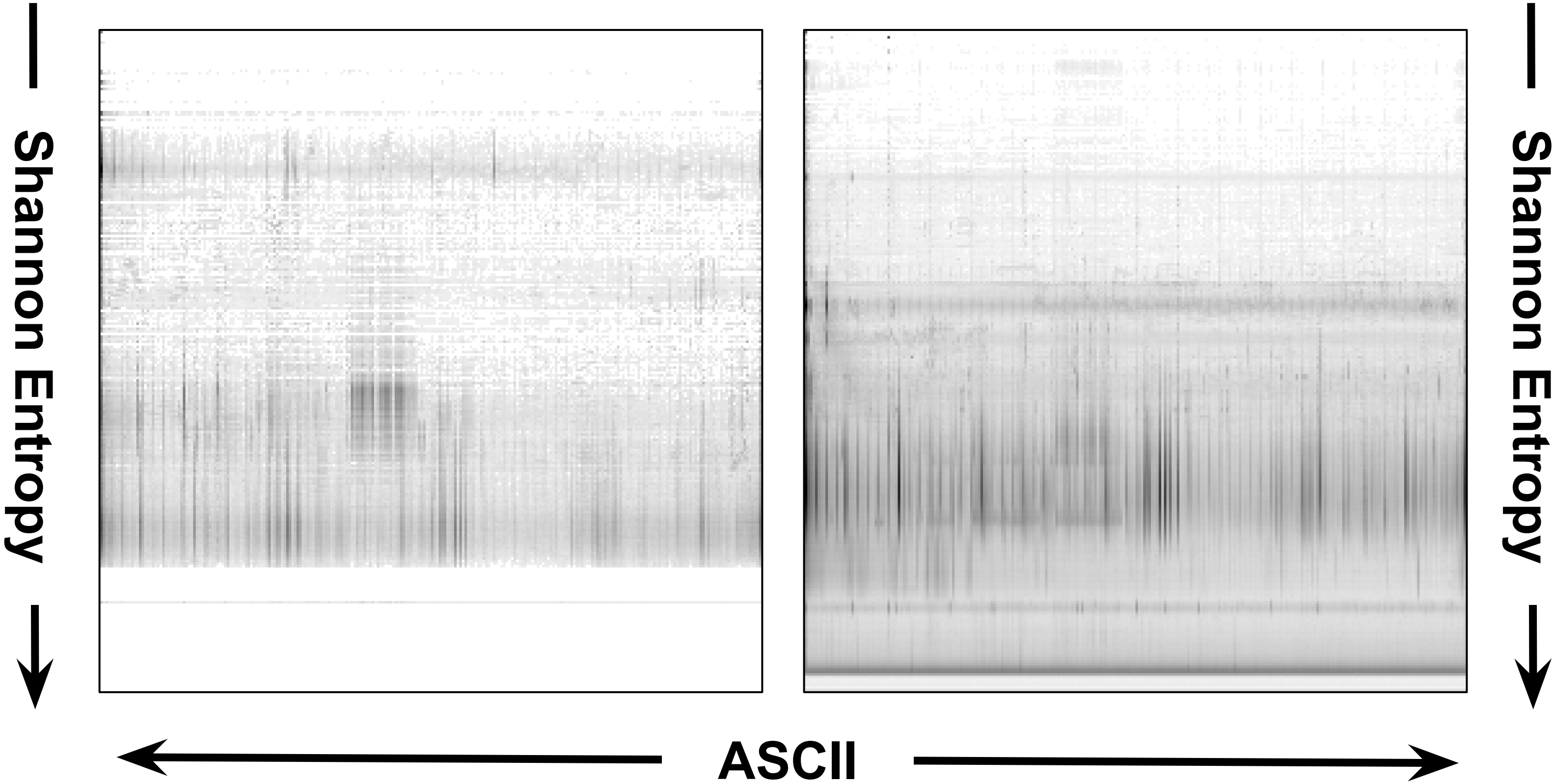
- Instruction N-grams
- Statistics
  - Function, Blocks
  - Calls, Branches

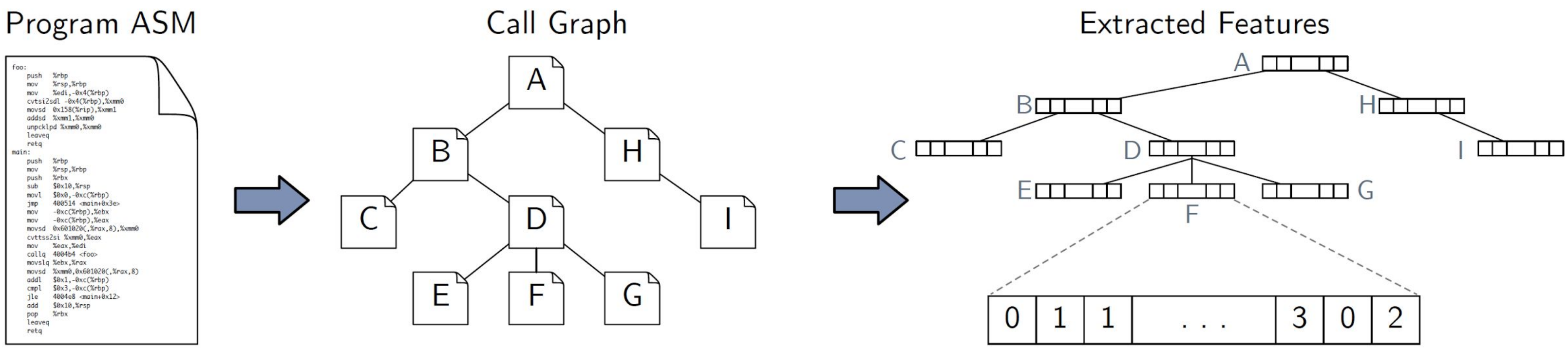
State-of-the-art: Bytes Analysis



[Saxe and Berlin, 2015] Deep neural network based malware detection using two dimensional binary program features.

State-of-the-art: Bytes Analysis

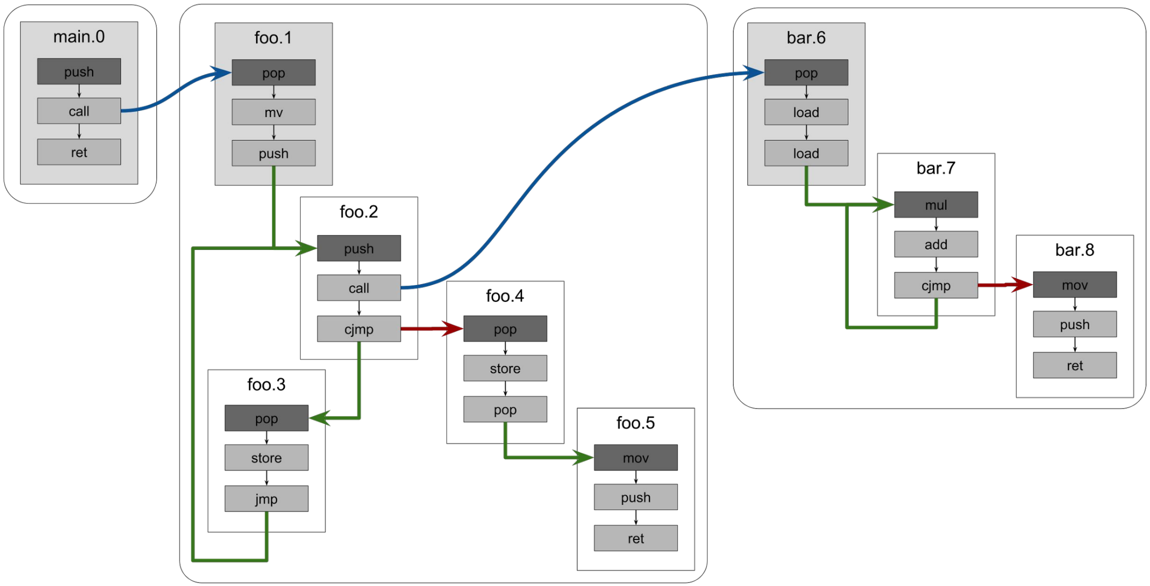




Reverse engineering: Radare2

Extract code from: x86, JAR, JS, etc

Compiler Internal Representation



Instruction Flow Graph

Control Flow Graph

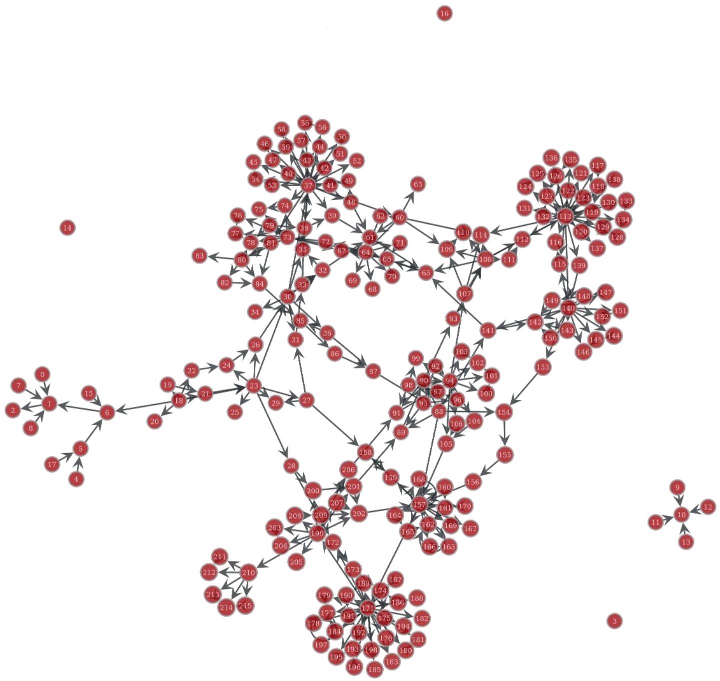
Call Graph

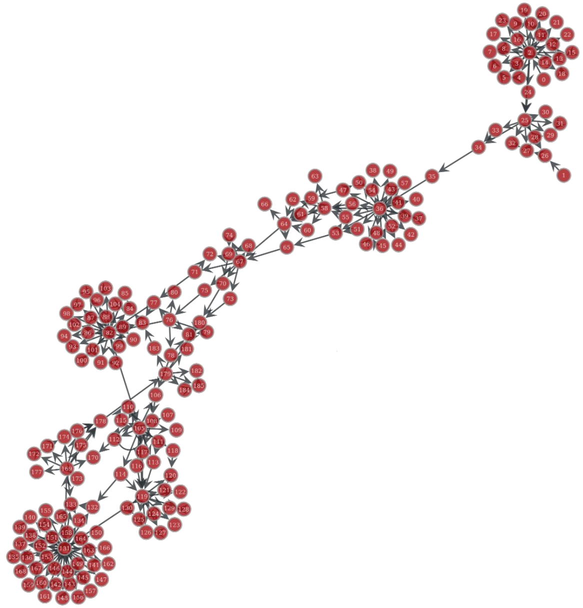
Statistics

Instructions 1-grams

Instructions 2-grams

Compiler-based Graphs

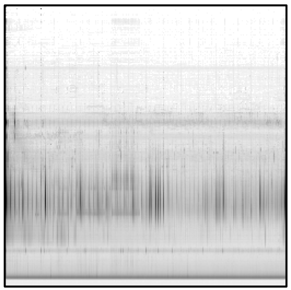




Malware Characterization

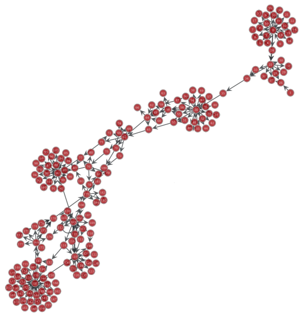
Bytes-Entropy Histogram

- Signature of the binary
- 256 x 256 positive integers
- Fast to extract (~ms)



Disassembled Code

- Multiple graphs
- No fixed size
- Long reverse engineering (~s)

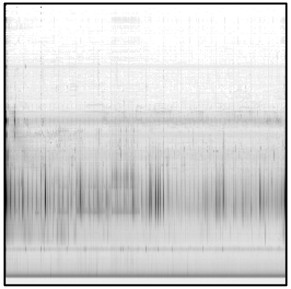


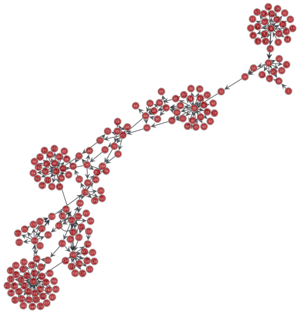
ML + Compiler-based Graphs for Malware

Malware Classification

Millions to classified
- Reversing Labs
- Virus Total

Models trained often



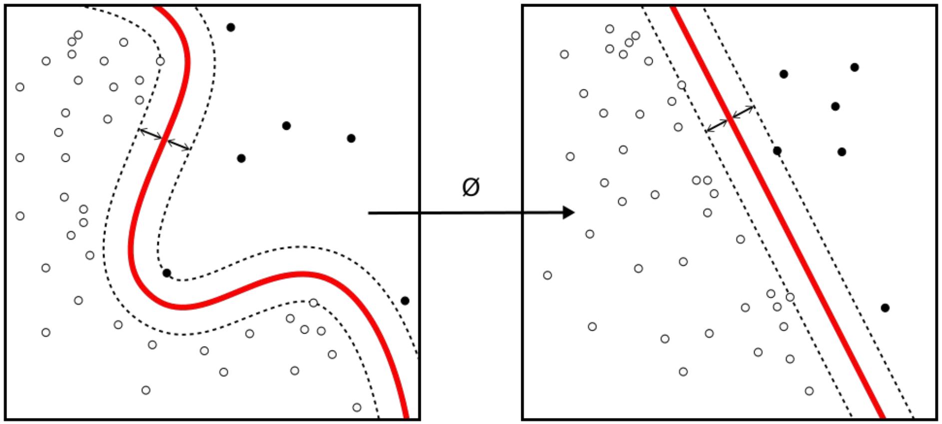


SVM and Graph Kernel

Support Vector Machine

- Linear Separator
- Kernel Trick
- Graph Kernels

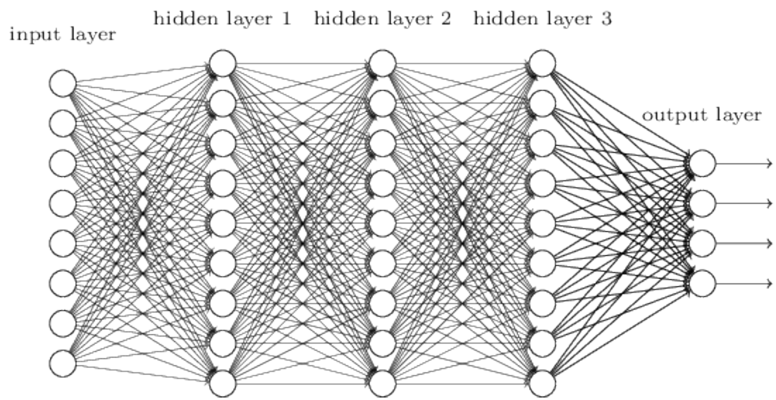
Problem: **Poor Scaling**

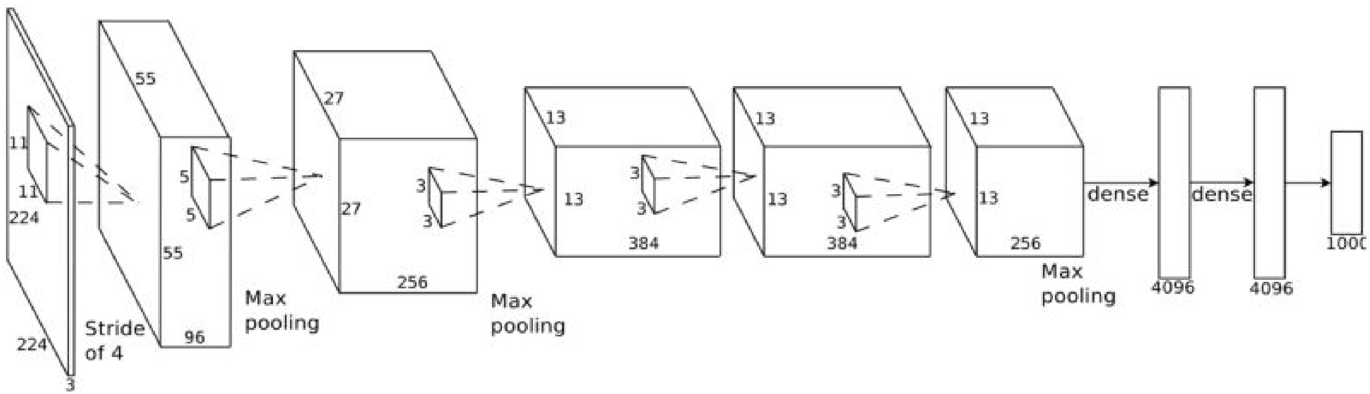


Deep Learning

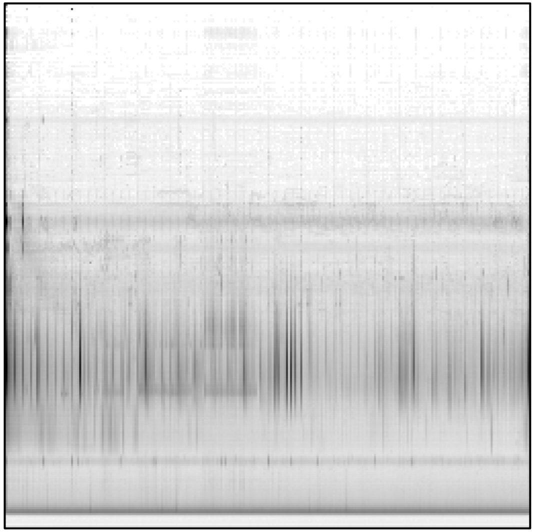
Old concept

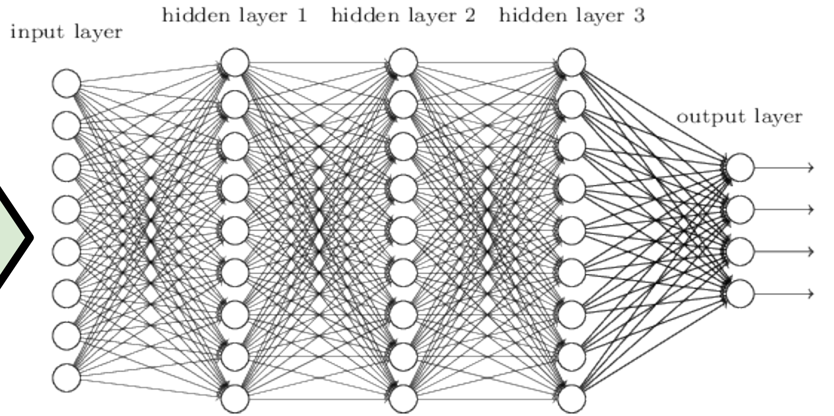
- Large model expensive
- Scale with dataset size
- Convolutional neural networks



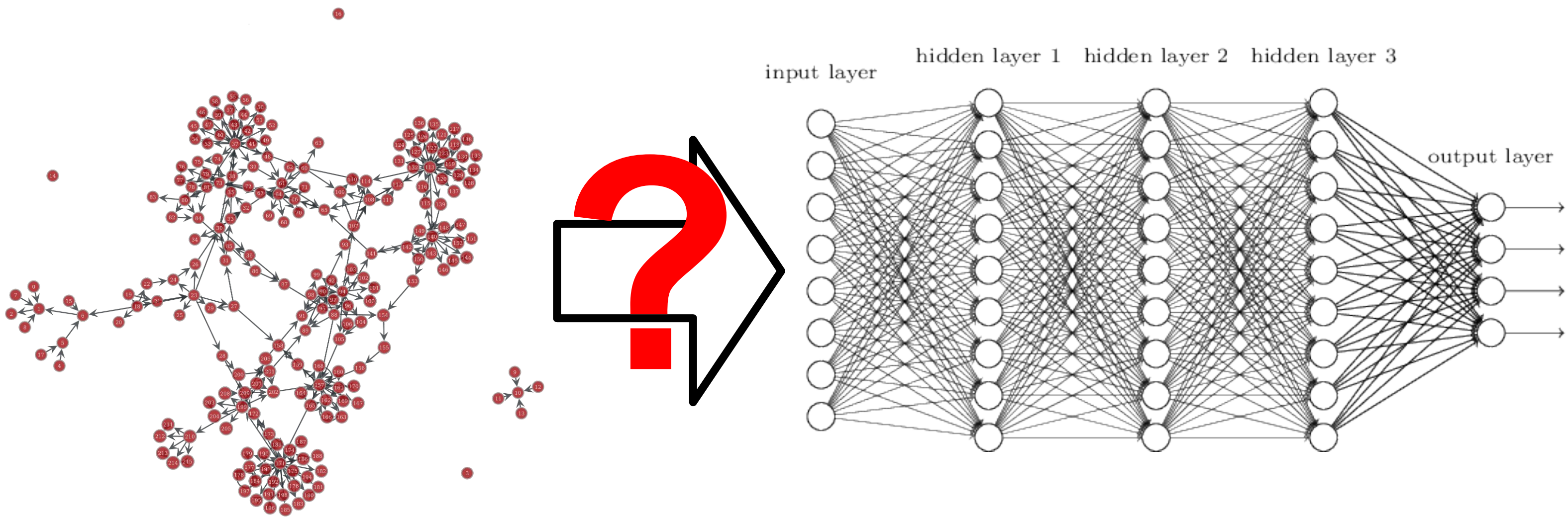


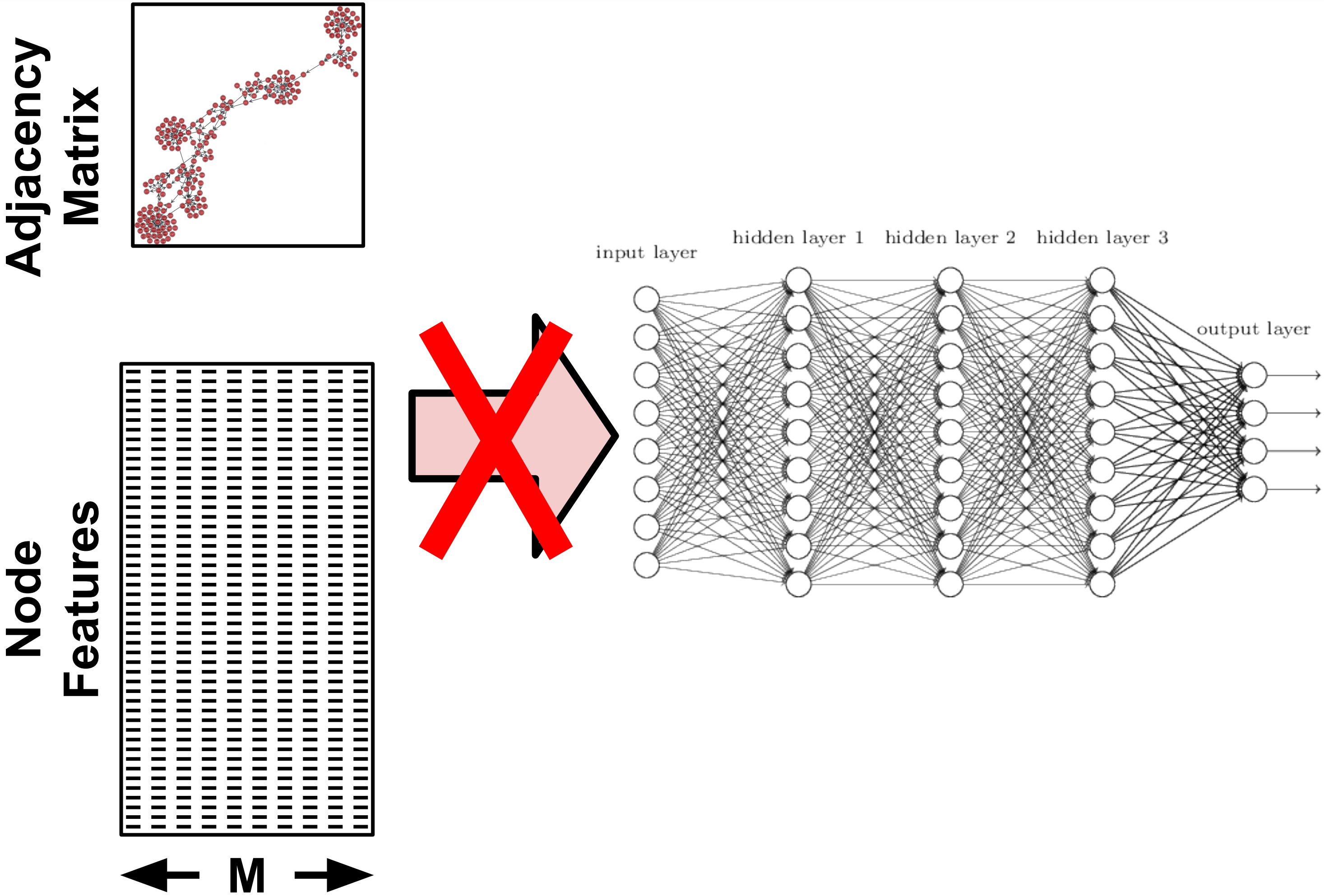
Deep Learning on Byte-Entropy Histogram





Deep Learning on Graphs

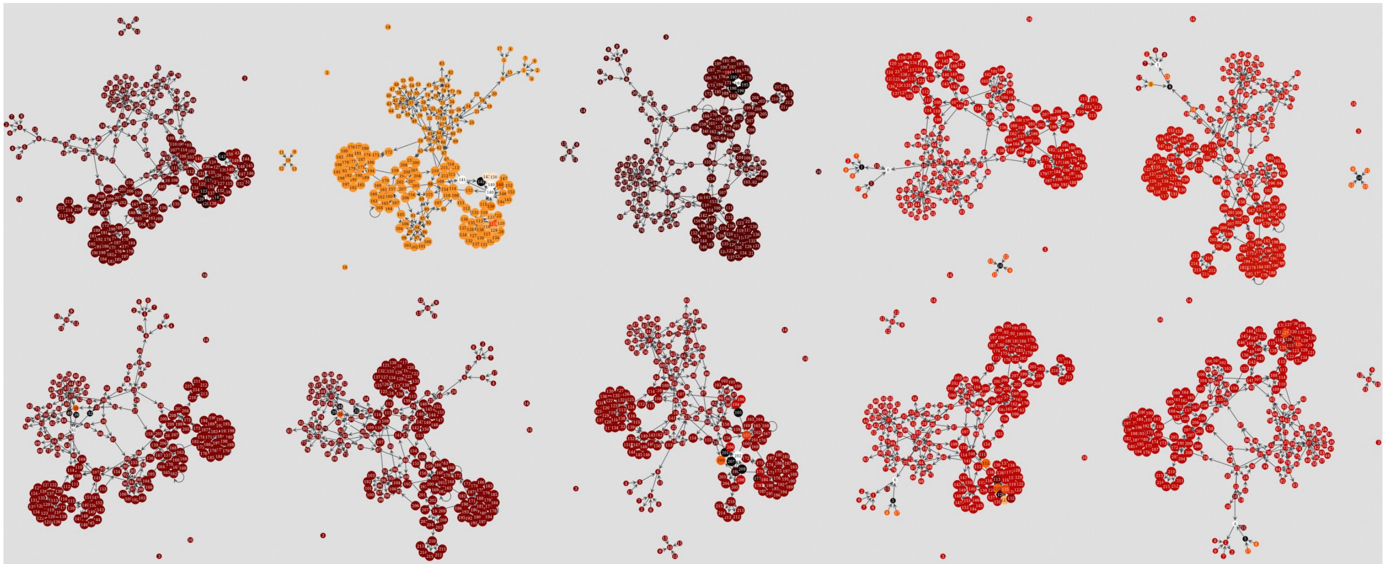


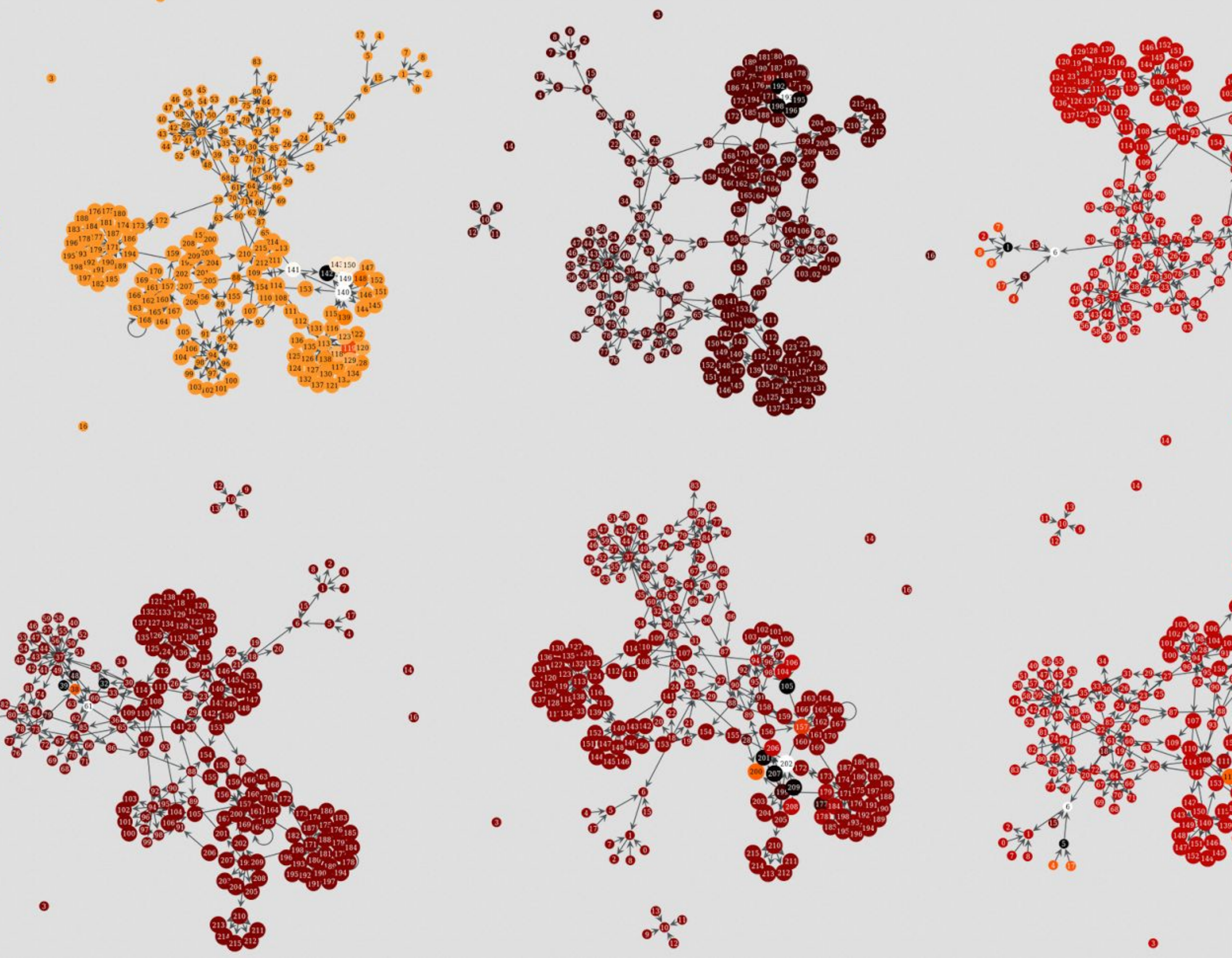

Adjacency
Matrix
Node
Features
M
input layer
hidden layer 1
hidden layer 2
hidden layer 3
output layer

Graph Spectral Analysis

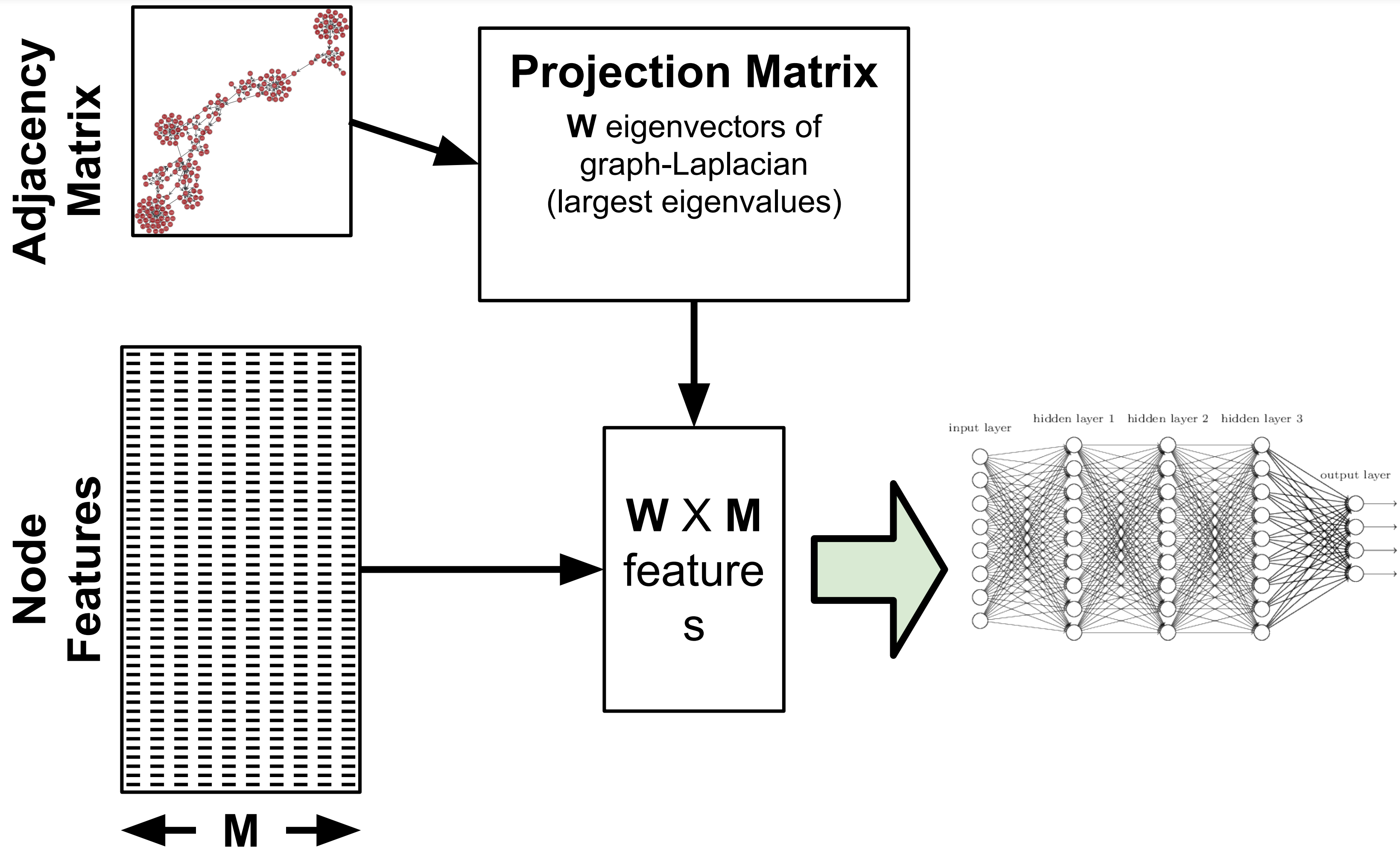
Eigenvectors of the graph-Laplacian

[Bronstein et al., 2016] *Geometric deep learning: going beyond euclidean data*

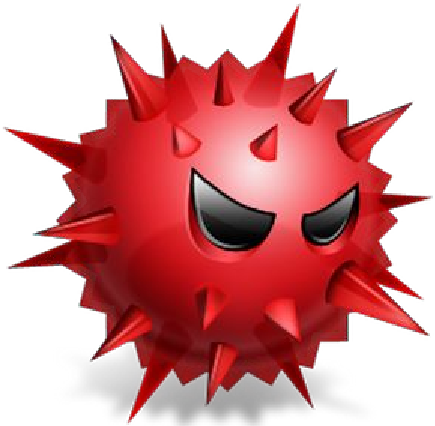




Graph Spectral Features



Malware Classification



Malware Stream

Reversing Labs

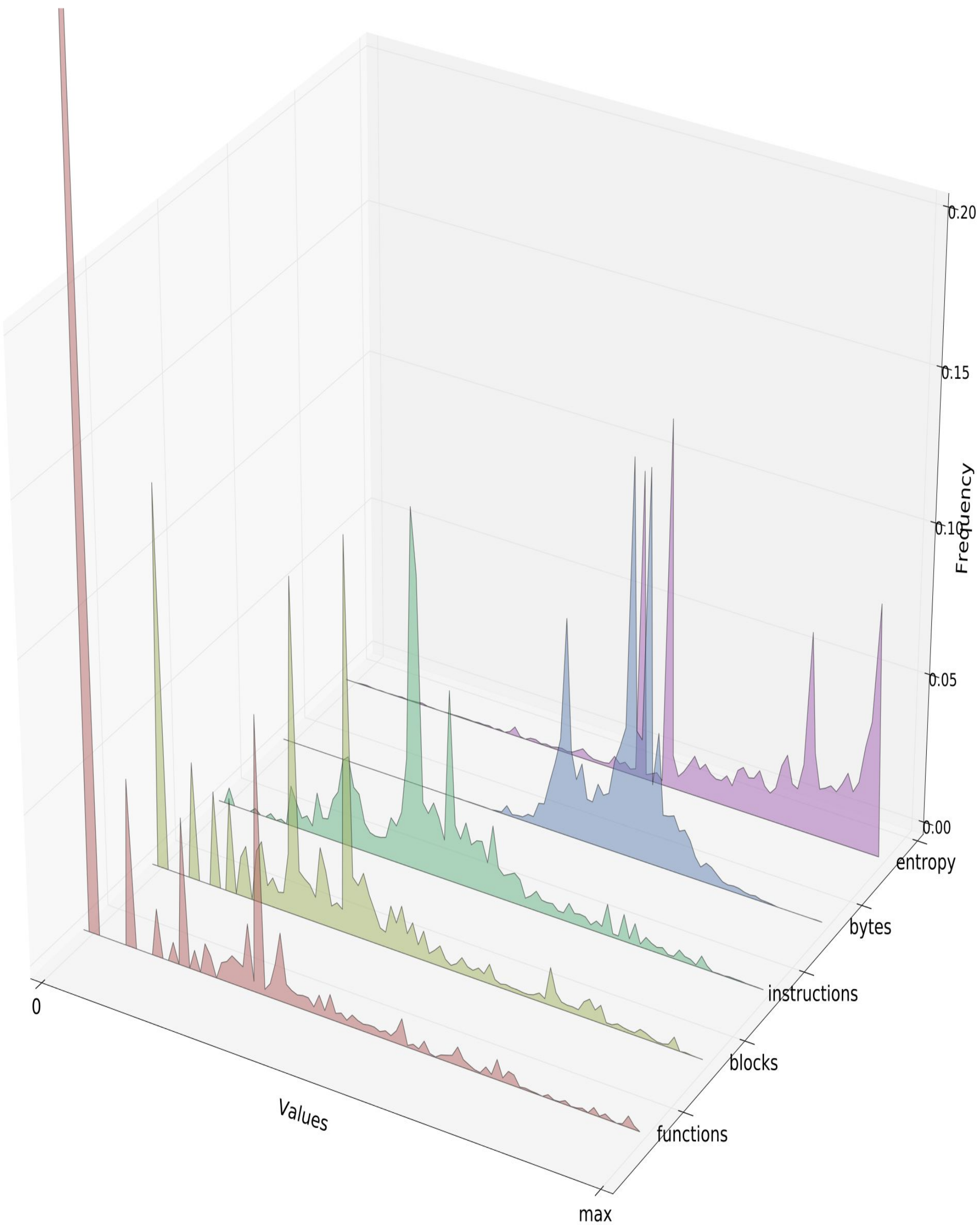
- Billions of malware
- Curated streams

Financial stream

- 1.2 millions
- 40+ families

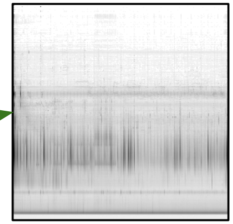
Selected

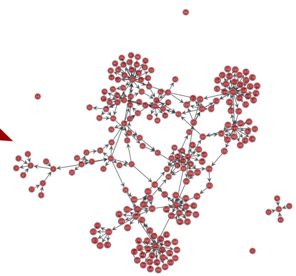
- families with more than 1,000 malware


0.20
0.15
0.10
0.05
0.00
Frequency
entropy
bytes
instructions
blocks
functions
0
Values
max

Malware Features for Deep Learning

Characterization		Format	Size
bytes-entropy	histogram	matrix	256 x 256
Global	statistics	vector	43
	1-grams		53
	2-grams		2809
Function	statistics	matrix	20 x 23
	1-grams		20 x 53
	2-grams		20 x 2809
Block	statistics	matrix	20 x 10
	1-grams		20 x 53
	2-grams		20 x 2809
Operations	statistic	matrix	20 x 2





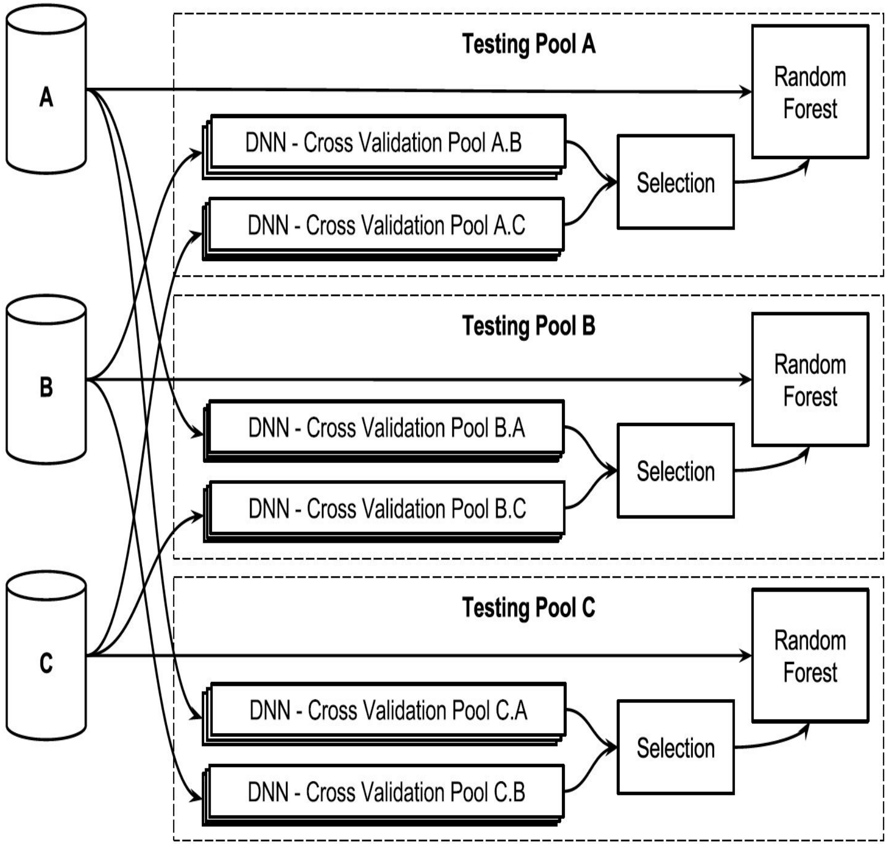
Characterization		Format	Size	DNN layers	DNN shape
bytes-entropy histogram		matrix	256 x 256	12	geometric
Global	statistics	vector	43	5	arithmetic
	1-grams		53	6	arithmetic
	2-grams		2809	9	geometric
Function	statistics	matrix	20 x 23	7	geometric
	1-grams		20 x 53	8	geometric
	2-grams		20 x 2809	12	geometric
Block	statistics	matrix	20 x 10	5	geometric
	1-grams		20 x 53	8	geometric
	2-grams		20 x 2809	12	geometric
Operations	statistic	matrix	20 x 2	5	arithmetic

Characterization		Format	Size	DNN layers	DNN shape
bytes-entropy histogram		matrix	256 x 256	12	geometric
Global	statistics	vector	43	5	arithmetic
	1-grams		53	6	arithmetic
	2-grams		2809	9	geometric
Function	statistics	matrix	20 x 23	7	geometric
	1-grams		20 x 53	8	geometric
	2-grams		20 x 2809	12	geometric
Block	statistics	matrix	20 x 10	5	geometric
	1-grams		20 x 53	8	geometric
	2-grams		20 x 2809	12	geometric
Operations	statistic	matrix	20 x 2	5	arithmetic

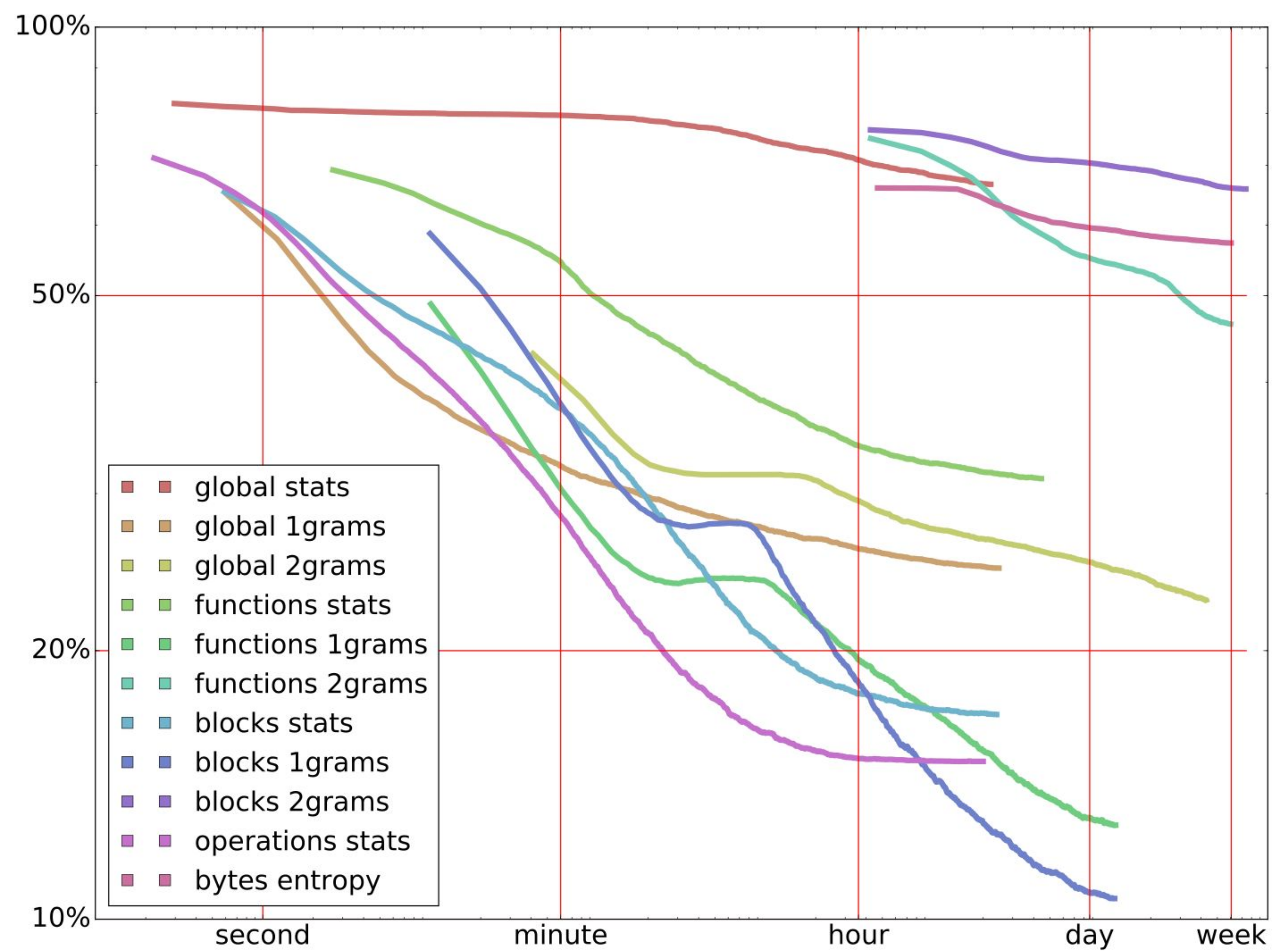
Cross-validation and Consensus

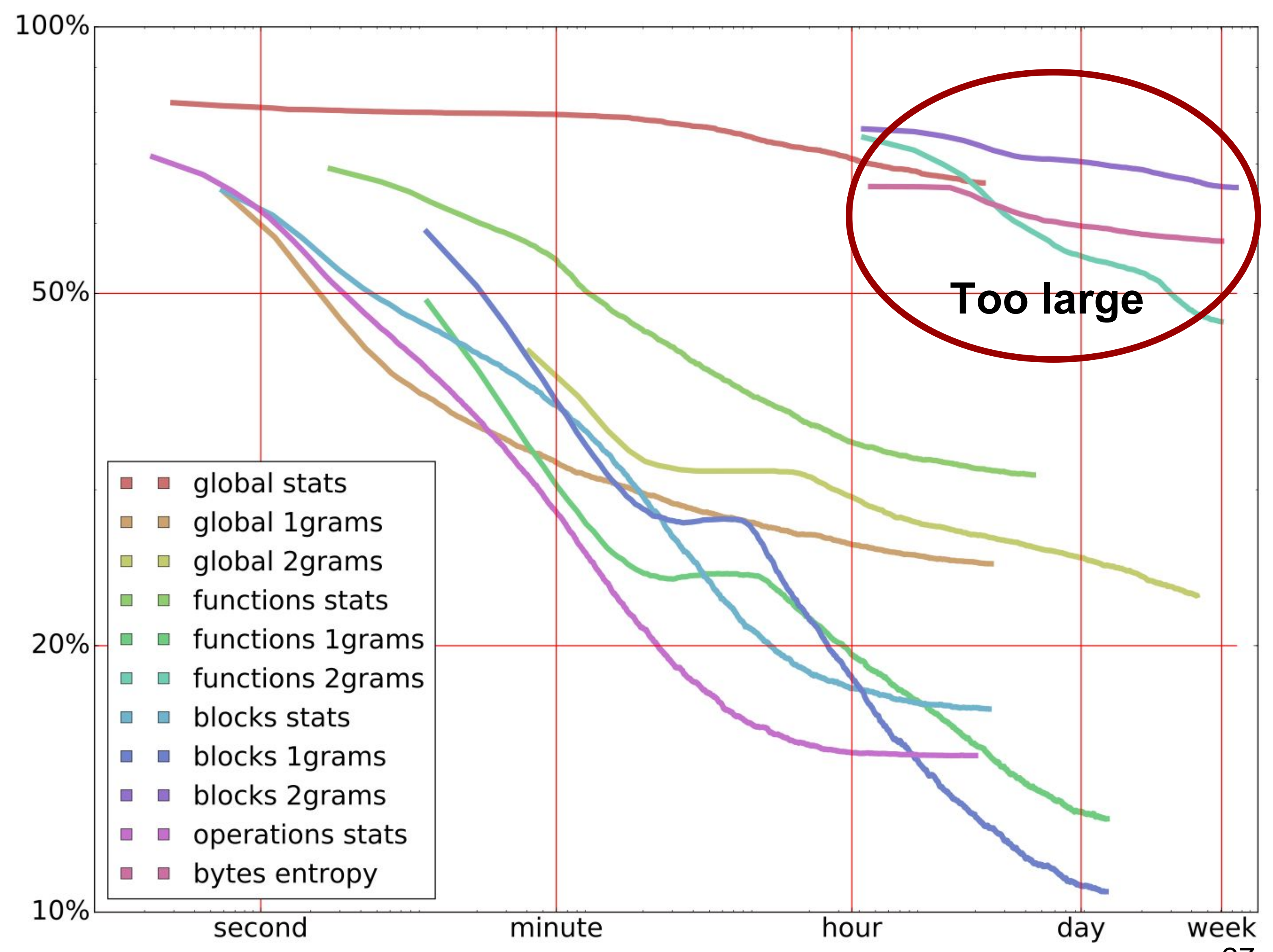
5-fold cross-validation

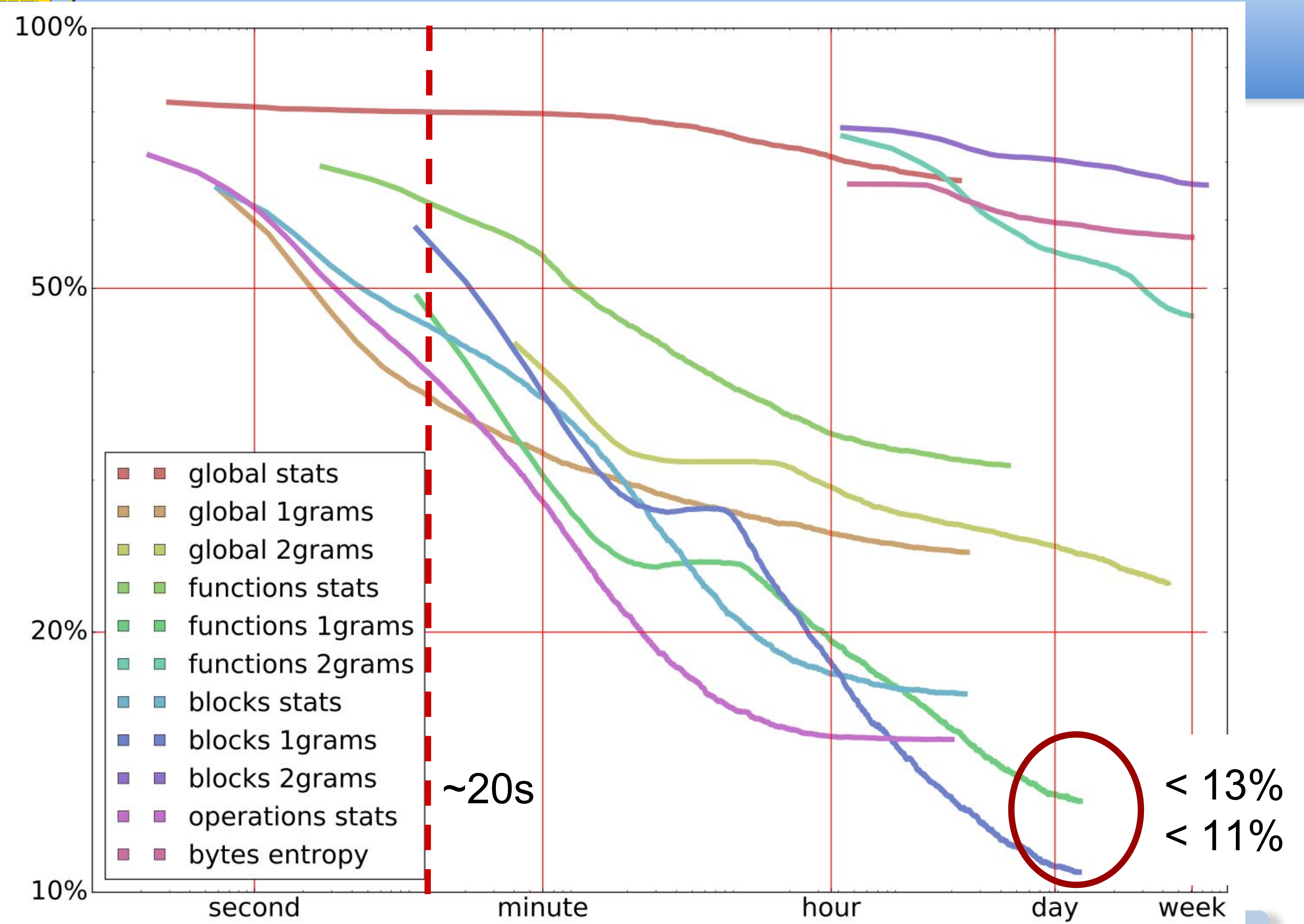
- Testing pools
- Cross-validation pool
  - Many DNNs
- Selection
  - K best DNNs
- DNN ensemble
  - Consensus



5 testing pools * 4 CV pools * 10 DNNs = **200 DNNs**


100%
50%
20%
10%
second
minute
hour
day
week
global stats
global 1grams
global 2grams
functions stats
functions 1grams
functions 2grams
blocks stats
blocks 1grams
blocks 2grams
operations stats
bytes entropy


100%
50%
20%
10%
second
minute
hour
day
week
global stats
global 1grams
global 2grams
functions stats
functions 1grams
functions 2grams
blocks stats
blocks 1grams
blocks 2grams
operations stats
bytes entropy
Too large


100%
50%
20%
10%
second
minute
hour
day
week
global stats
global 1grams
global 2grams
functions stats
functions 1grams
functions 2grams
blocks stats
blocks 1grams
blocks 2grams
operations stats
bytes entropy
~20s
< 13%
< 11%

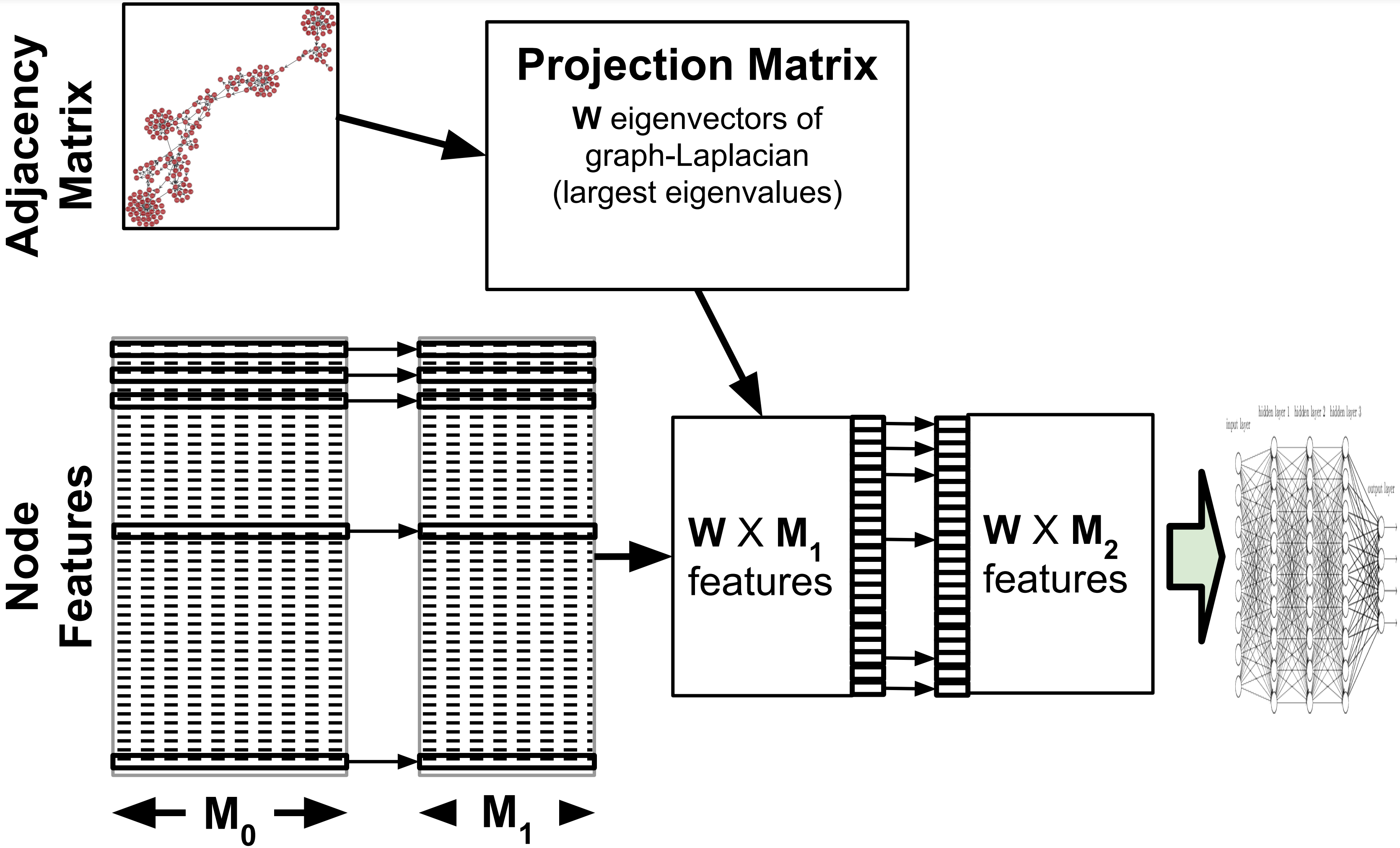
Models		1 best	2 best	5 best
bytes-entropy histogram		39.2%	32.8%	27.3%
Global	statistics	60.5%	53.2%	44.4%
	1-grams	22.9%	20.8%	19.1%
	2-grams	20.7%	19.2%	18.5%
Function	statistics	26.5%	23.8%	20.3%
	1-grams	12.3%	12.0%	10.8%
	2-grams	30.5%	27.2%	25.7%
Block	statistics	15.8%	14.8%	13.6%
	1-grams	10.3%	9.9%	9.9%
	2-grams	39.2%	35.1%	31.4%
Operations	statistic	13.6%	11.3%	10.4%

Best Results: DNN Ensemble

Models	1 best	2 best	5 best
BEH & Global level	16.0%	15.3%	13.8%
Compiler Graphs	8.4%	8.0%	8.0%
All Features	6.9%	6.5%	6.3%

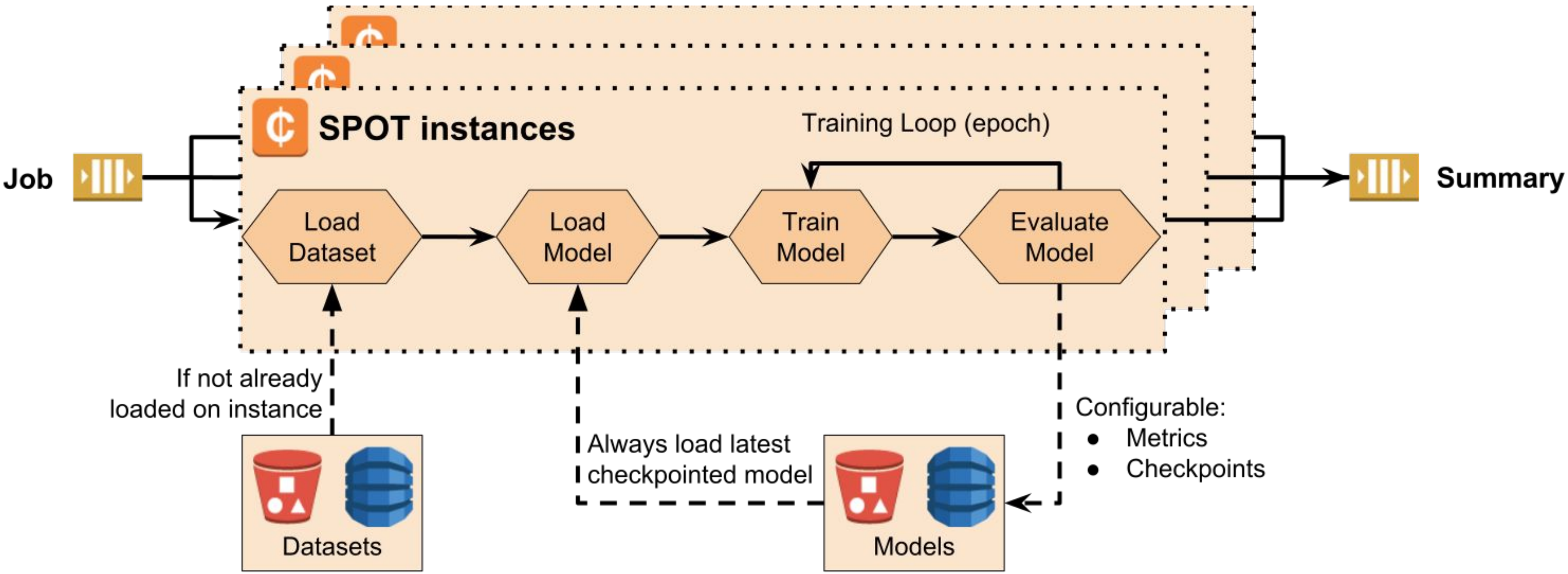
More than twice as good when adding Compiler-based Graphs

Improving Deep Learning Model



New DNN model

- Less parameters to learn
  - Train on full malware dataset



Dataset for the class

- Files
  - ***financial-1000***
  - ***All ?***

- New features
  - ASCII strings
  - More classifications sources (VirusTotal)

- Lessons learned
  - Bytes-Entropy Histograms
    - 16 entropy bins (16x256)
    - Different sliding windows **(** 1024 by 256 **and** 2048 by 128 **)**
  - Assembly Graphs
    - Removed 2-grams
    - 1-grams at operation level

QUESTIONS ?

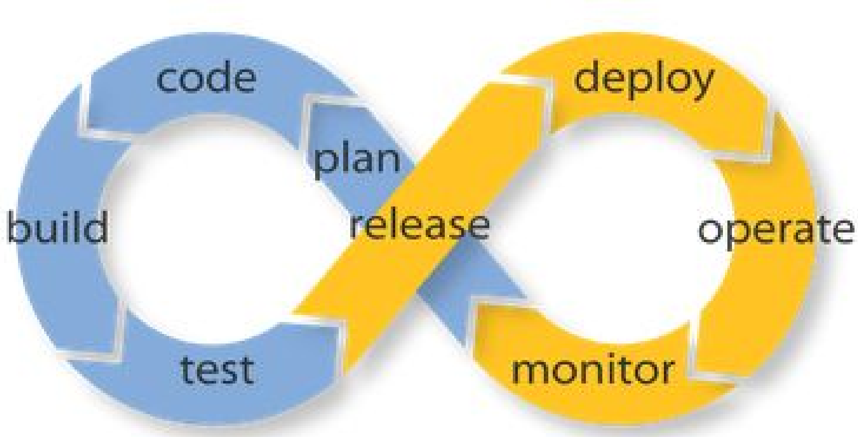
Characterization

Deep-learning

Datasets

Results

CISC850: DevOps



Tristan Vanderbruggen

Dept of Computer & Information Sciences

University of Delaware

Tools

- Communication
  - Slack
- Version Control
  - GitHub
  - git
- Scrum
  - Waffle I/O
- Cloud
  - EC2
  - S3

Teams

- Analysis 1
  - Leonardo De La Rosa
  - Yang Yang
  - Yuhao Peng
- Analysis 2
  - Sean Kilgallon
  - Ashwag Altayyar
  - Peng Su
- Chatbot
  - Aman Sawhney
  - Abhijeet Srivastava
  - Anupam Basu
- Graphs
  - Ian Lantzy
  - Fan Li
  - Paul Soper
- Visual Analytics 1
  - Wanxin Li
  - Zicheng Liu
  - Ezeanaka Kingsley
  - Abdulrahaman Alshammari
- Visual Analytics 2
  - Yujun Zeng
  - Ruikai Zheng
  - Hancheng Zhao
  - Ruijie Xi
- Machine Learning
  - Vinit Singh
  - Abhilash Parthasarathy
  - Mingxing Gong

Communication

- Slack
  - https://ud-cisc850.slack.com/signup
  - Team lead create channel
    - Invite other members
    - Invite Dr Cavazos
  - Send me **private message** with:
    - name
    - project
    - **GitHub** username

Version Control & Scrum

- Version Control: GitHub
  - https://github.com/cavazos-lab/
  - Find repo for your group
    - `spring-2017-CISC850-XXX`
- Scrum: Waffle IO
  - https://waffle.io/cavazos-lab/spring-2017-CISC850-XXX
  - Create backlogs:
    - Documentations
    - Project design

Cloud

- Start instance
- SSH in the instance
- git CLI
  - `sudo yum install git vim emacs`
  - `git clone https://github.com/cavazos-lab/XXX XXX`
  - `cd XXX`
  - `vim README.md`

    `Make changes: each member fill a section`
  - `git status`
  - `git add README.md`
  - `git commit -m "update README with ..."`
  - `git push origin master`
  - **???**
- More on GIT:
  - GIT Book: https://git-scm.com/book/en/v2

QUESTIONS ?

AWS

GIT

Scrum