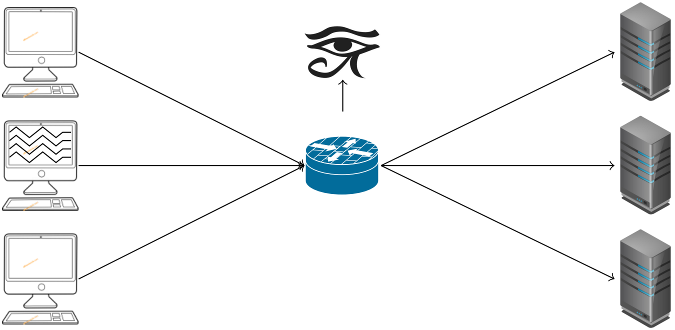
Structured models with neural networks

Tomáš Pevný

February 28, 2020

Motivation — identification of infected computers



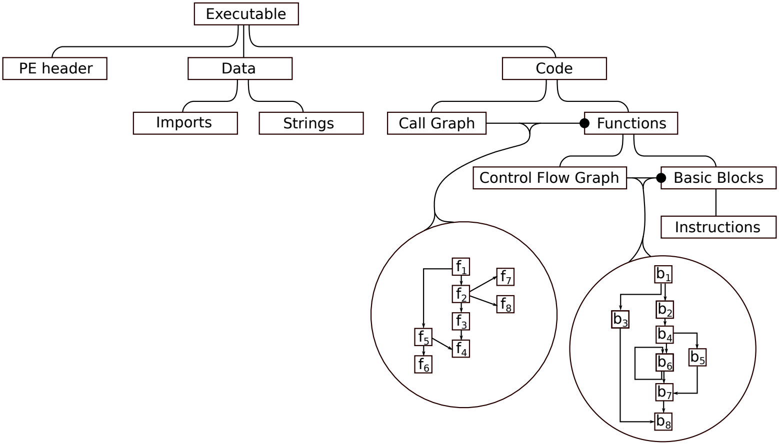
examples of messages:

http://lop.guardpair.com/affs?addonname=[Enter%20Product%20Name]
&affid=9050&subaffid=5774&subID=undefined&clientuid=undefined
&origaffid=9050&origsubaffid=5774&href=http%3A%2F%2F7

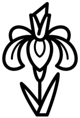
http://pf.updatewp.org/?v=3.16&pcrc=308403836&LSVRDT=&ty=CHECK

http://rules.similardeals.net/v1.0/whitelist/1052/9050x5774/
7online.subsea7.net?partnerName=Lyrics

Motivation — representation of PE file format



Single-instance learning



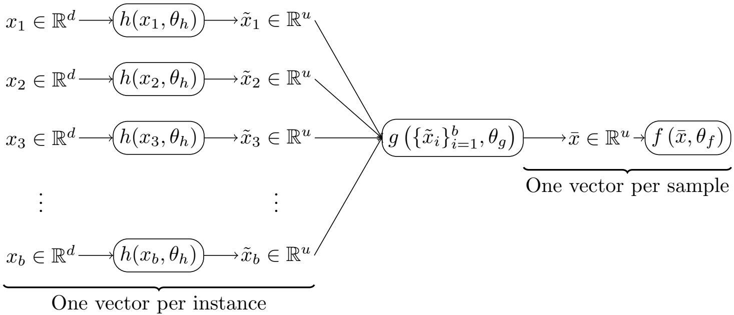
$$\xrightarrow[\text{features}]{\text{extract}} x = (x_1, \ldots, x_d) \xrightarrow[\text{classifier}]{\text{send to}} f(x) = \begin{cases} +1 \\ -1 \end{cases}$$

Icon made by Freepik from www.flaticon.com

Multi-instance learning

$$\xrightarrow[\text{features}]{\text{extract}} x = \left\{ \begin{matrix} (x_{1,1}, \ldots, x_{1,d}) \\ (x_{1,1}, \ldots, x_{1,d}) \\ \vdots \\ (x_{b,1}, \ldots, x_{b,d}) \end{matrix} \right\} \xrightarrow[\text{classifier}]{\text{send to}} f(x) = \begin{cases} +1 \\ -1 \end{cases}$$

Our solution of multi-instance learning



$$h(x, \theta_h) : \mathbb{R}^d \to \mathbb{R}^u$$
$$h(x, \theta_h) = \max\{0, x^{\mathrm{T}}\theta_h\}$$

$$g\left(\{\tilde{x}_i\}_{i=1}^b\right) = \frac{1}{l}\sum_{i=1}^b \tilde{x}_i$$

$$f(\bar{x}, \theta_f) : \mathbb{R}^u \to \mathbb{R}^2$$
$$f(\bar{x}, \theta_f) = \bar{x}^{\mathrm{T}}\theta_f$$

Generalized multi-instance learning

$$\xrightarrow[\text{features}]{\text{extract}} x = \left\{ \begin{array}{c} (x_1^{(1)}, \ldots, x_{d_1}^{(1)}) \\ \\ \left\{ \begin{array}{c} (x_{1,1}^{(2)}, \ldots, x_{1,d_2}^{(2)}) \\ (x_{1,1}^{(2)}, \ldots, x_{1,d_2}^{(2)}) \\ \vdots \\ (x_{b,1}^{(2)}, \ldots, x_{b,d_2}^{(2)}) \end{array} \right\} \\ \\ \left\{ \begin{array}{c} (x_{1,1}^{(3)}, \ldots, x_{1,d_3}^{(3)}) \\ (x_{1,1}^{(3)}, \ldots, x_{1,d_3}^{(3)}) \\ \vdots \\ (x_{c,1}^{(3)}, \ldots, x_{c,d_3}^{(3)}) \end{array} \right\} \end{array} \right\} \xrightarrow[\text{classifier}]{\text{send to}} f(x) = \begin{cases} +1 \\ -1 \end{cases}$$

Extension of universal approximation theorem

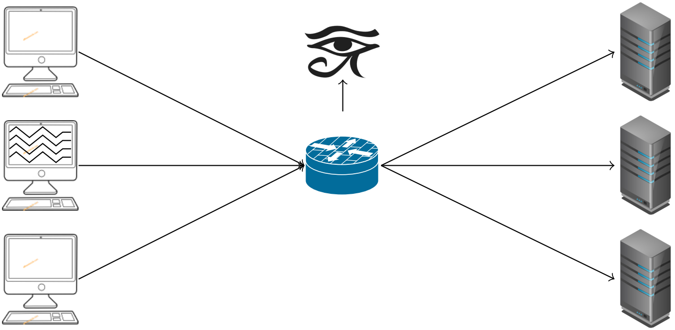
Let $\mathcal{S}$ be the class of spaces which

1. contains all compact subsets of $\mathbb{R}^d$, $d \in \mathbb{N}$
2. is closed under finite cartesian products
3. for each $\mathcal{X} \in \mathcal{S}$ we have $\mathcal{P}(\mathcal{X}) \in \mathcal{S}$[1]

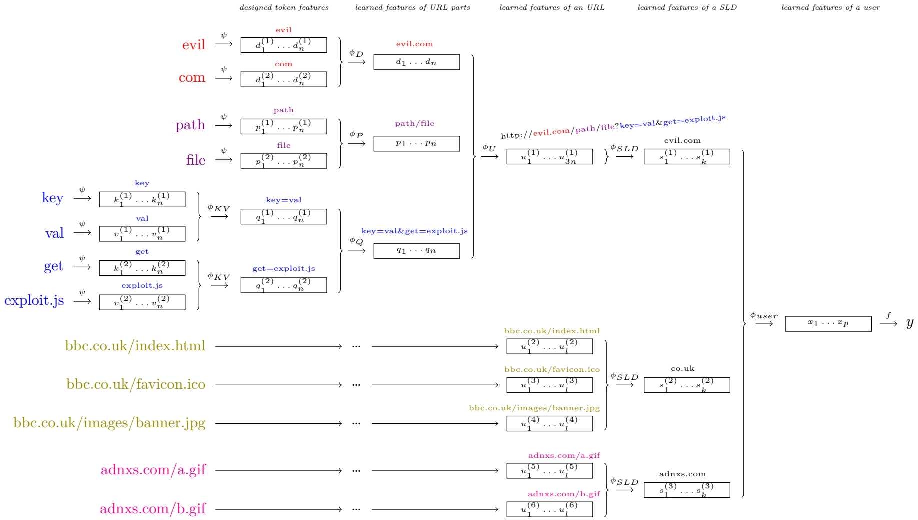
Then for each $\mathcal{X} \in \mathcal{S}$, every continuous function on $\mathcal{X}$ can be arbitrarily well approximated by neural networks.

[1] Here we assume that $\mathcal{P}(\mathcal{X})$ is endowed with some metric.

Identification of infected computers

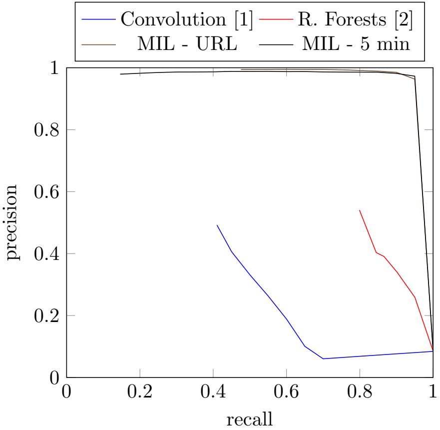


Detection of infected users — model



Detection of infected users — results

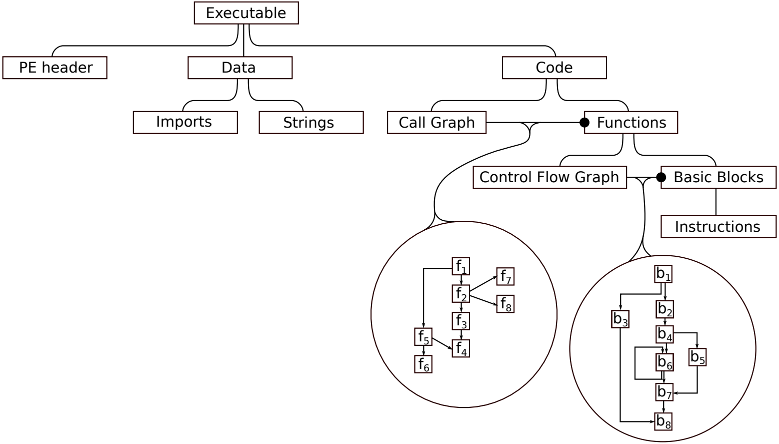
- Training data from 5-30.10 2017
  $3.6 \cdot 10^9 / 4.45 \cdot 10^6$ urls / users
- Testing data from 3.11 2017
  $2 \cdot 10^8 / 1.5 \cdot 10^6$ urls / users



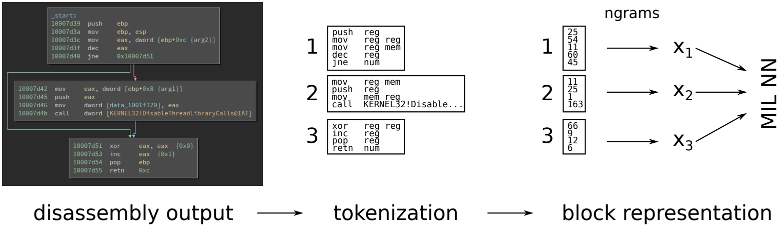
[1] *eXpose: A character-level convolutional neural network with embeddings for detecting malicious URLs*, file paths and registry keys, J. Saxe and K. Berlin, 2017

[2] *Learning detectors of malicious web requests for intrusion detection in network traffic*, L. Machlica, K. Bartos, and M. Sofka, 2017

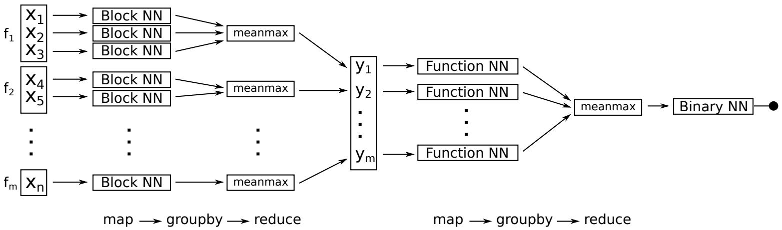
Static analysis of malware binaries — PE file format



Static analysis of malware binaries — model of a function

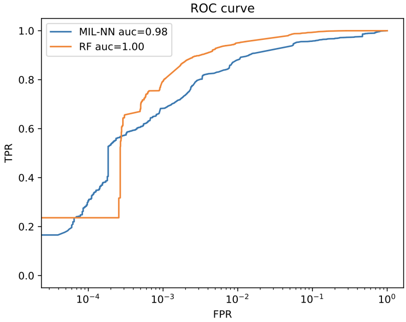


Static analysis of malware binaries — model of a binary



Static analysis of malware binaries — results

- Training data
  $4 \cdot 10^5$ binaries
- Testing data
  $5.5 \cdot 10^5$ binaries
- approx. 10% were malicious



Static analysis of malware binaries — feedback

- avoids specifying imports in PE header
- loads addresses of library functions to an array

```
push    ebx {var_4}  {0x0}
mov     ebx, dword [KERNEL32!LoadLibraryA@IAT]
push    esi {var_8}
push    edi {var_c}
push    dword [data_47f274] {var_10}
call    ebx
push    dword [data_47f278] {var_14}
mov     esi, dword [KERNEL32!GetProcAddress@IAT]
mov     edi, eax
push    edi {var_18}
call    esi
push    dword [data_47f27c] {var_1c}
mov     dword [data_481378], eax
push    edi {var_20}
call    esi
push    dword [data_47f280] {var_24}
mov     dword [data_48137c], eax
push    edi {var_28}
call    esi
push    dword [data_47f284] {var_2c}
mov     dword [data_481380], eax
push    edi {var_30}
call    esi
push    dword [data_47f288] {var_34}
mov     dword [data_481384], eax
push    edi {var_38}
call    esi
```

Static analysis of malware binaries — feedback

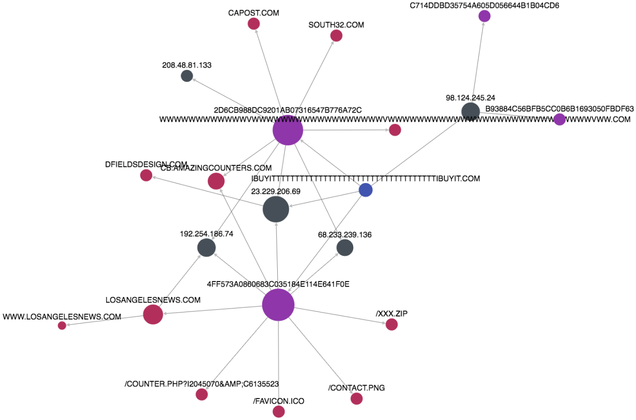
- adware related behavior
- creating both visible and invisible windows

```
mov     eax, dword [edi+0x38]
push    0x7f00 {var_7c_1}
push    0x0 {var_80_1}
mov     dword [esp+0x14 {var_6c_1}], 0x30
mov     dword [esp+0x18 {var_68_1}], 0x3
mov     dword [esp+0x1c {var_64_1}], 0x402d00
mov     dword [esp+0x20 {var_60_1}], 0x0
mov     dword [esp+0x24 {var_5c}], 0x0
mov     dword [esp+0x28 {var_58_1}], eax
mov     dword [esp+0x2c {var_54_1}], 0x0
call    dword [USER32!LoadCursorW@IAT]
mov     dword [esp+0x28 {var_58_2}], eax
lea     eax, [esp+0xc {var_74}]
push    eax {var_74} {var_84}
mov     dword [esp+0x30 {var_54_2}], 0x9
mov     dword [esp+0x34 {var_50_1}], 0x0
mov     dword [esp+0x38 {var_4c_1}], 0x4553b4
mov     dword [esp+0x3c {var_48_1}], 0x0
call    dword [USER32!RegisterClassExW@IAT]
mov     word [data_475ba4], ax
test    ax, ax
jne     0x40252a
```

Future directions

- Learning representation of hierarchical data
  - Learning distances for clustering
  - Generative models
  - Anomaly / few shot learning
- Game theory in security
  - Finding new attacks under constraints
- Decentralized learning
- Modeling and reasoning over relational data

Modeling the internet



https://www.threatcrowd.org/domain.php?domain=ibuyittttttttttttttttttttttttttttttttttttttibuyit.com

Materials

- `https://github.com/pevnak/Mill.jl`
- `https://github.com/pevnak/JsonGrinder.jl`
  - Discriminative models for multi-instance problems with tree-structure, Tomáš Pevný, Petr Somol, 2016
  - Using Neural Network Formalism to Solve Multiple-Instance Problems, Tomáš Pevný, Petr Somol, 2016
  - Approximation capability of neural networks on sets of probability measures and tree-structured data, Tomáš Pevný, Vojtěch Kovařík, 2019
  - Nested Multiple Instance Learning in Modelling of HTTP network traffic, Tomas Pevny, Marek Dedic, 2020