TRZYDZIESTE FORUM TELEINFORMATYKI®

Analiza efektywności wybranych metod grafowego uczenia maszynowego do detekcji złośliwego oprogramowania

Opiekun pracy: dr hab. inż. Zbigniew Tarapata, prof. WAT
Autor: mgr inż. Jan Romańczuk

WAT

WYDZIAŁ CYBERNETYKI WAT

Cele pracy

Przegląd

- Wykonanie przeglądu metod GML
- Wykonanie przeglądu metod GML do detekcji złośliwego oprogramowania
- Zdefiniowanie miar oceny klasyfikatorów złośliwego oprogramowania

Klasyfikacja

- Wybór innowacyjnych i interesujących metod grafowego uczenia maszynowego do detekcji złośliwego oprogramowania z uzasadnieniem

Implementacja

- Zaproponowanie i teoretyczny opis usprawnień wybranej metody GML do detekcji złośliwego oprogramowania
- Implementacja wybranych metod GML do tego celu

Opracowanie

- Badanie własności zaimplementowanych metod z wykorzystaniem zdefiniowanych miar oceny na zbiorze danych testowych

Cele nadrzędne

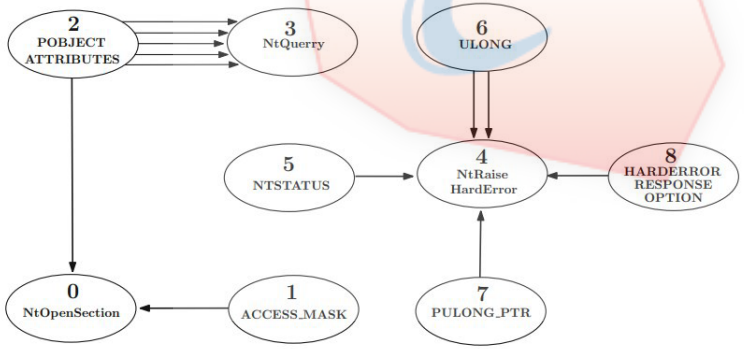
Rozbudowanie programu badawczego z przestudiowanego artykułu (A. Oliveira i R. J. Sassi, „Behavioral Malware Detection Using Deep Graph Convolutional Neural Networks,” *International Journal of Computer Applications 174/29,* 2021)

Implementacja nowych modeli realizujących zaproponowane usprawnienia na podstawie pierwotnego programu

Analiza porównawcza zaimplementowanych modeli

Przeszukiwanie bazy danych zaklasyfikowanych już próbek w postaci diagramów CFG (ang. $Control\ Flow\ Graph\ -\ graf\ przepływu\ sterowania$)

Porównywanie próbek zaklasyfikowanych z próbką badaną następuje poprzez wyznaczenie odległości edycji grafu ($Graph\ Edit\ Distance$).



Przykładowy graf wywołań funkcji systemowych podejrzanego programu. Może być określony jako wysokopoziomowy CFG

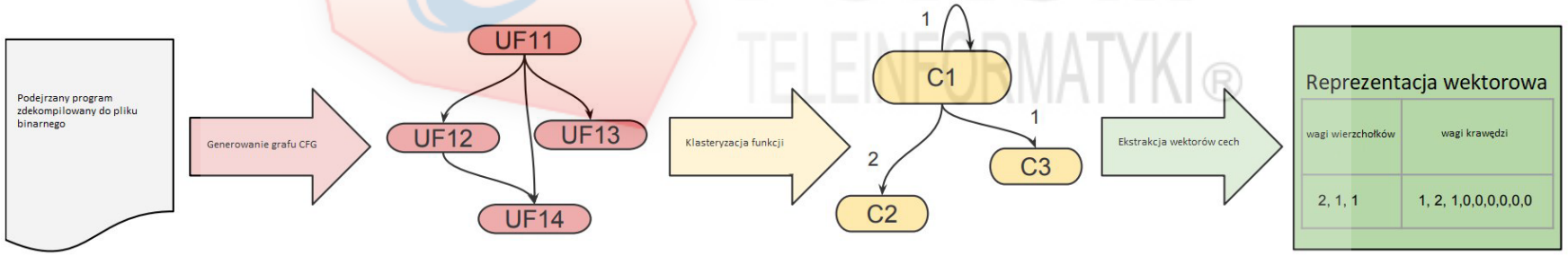
Źródło:
„Nikolopoulos S., Polenakis I., "A graph-based model for malware detection and classification using system-call groups", Journal of Computer Virology and Hacking Techniques, Volume 13, pages 29–46, (2017).".
Źródło rysunku: „Hu X., Chiueh T., Shin K., „Large-Scale Malware Indexing Using Function-Call Graphs" Proceedings of the 16th ACM conference on Computer and communications security, pp. 611–620, (2009)."

Detekcja złośliwego oprogramowania z wykorzystaniem metody lasu losowego

Klasteryzacja (spajanie sąsiednich wierzchołków o podobnych cechach w jeden) grafów CFG w wysokopoziomowe grafy skierowane.

Ekstrakcja cech wysokopoziomowych grafów do postaci tabelarycznej (wektory zanurzeń)

Klasyfikacja wektorów zanurzeń z wykorzystaniem klasyfikatora lasu losowego, który przyporządkowuje rozkład prawdopodobieństwa przyporządkowania embeddingu do każdej z klas złośliwego oprogramowania



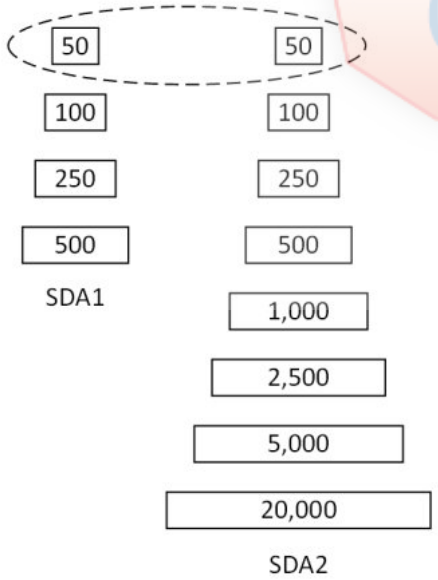
Ciąg przetwarzania cech funkcji wywoływanych przez podejrzane oprogramowanie

Źródło: „Hassen M., Chan P., „Scalable Function Call Graph-based Malware Classification", Proceedings of the 7th ACM Conference on Data and Application Security and Privacy, pp. 239-248, (2017)."

Wygenerowanie niskowymiarowych osadzeń grafowych próbek z wykorzystaniem metody $node2vec$ przez dwa równoległe enkodery

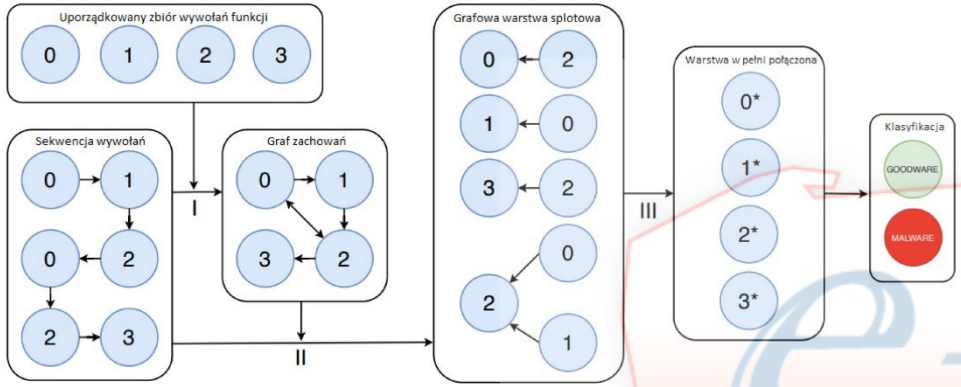
Agregacja wyjściowych osadzeń w jeden wektor wynikowy

Normalizacja wyniku funkcją aktywacji $ReLu$



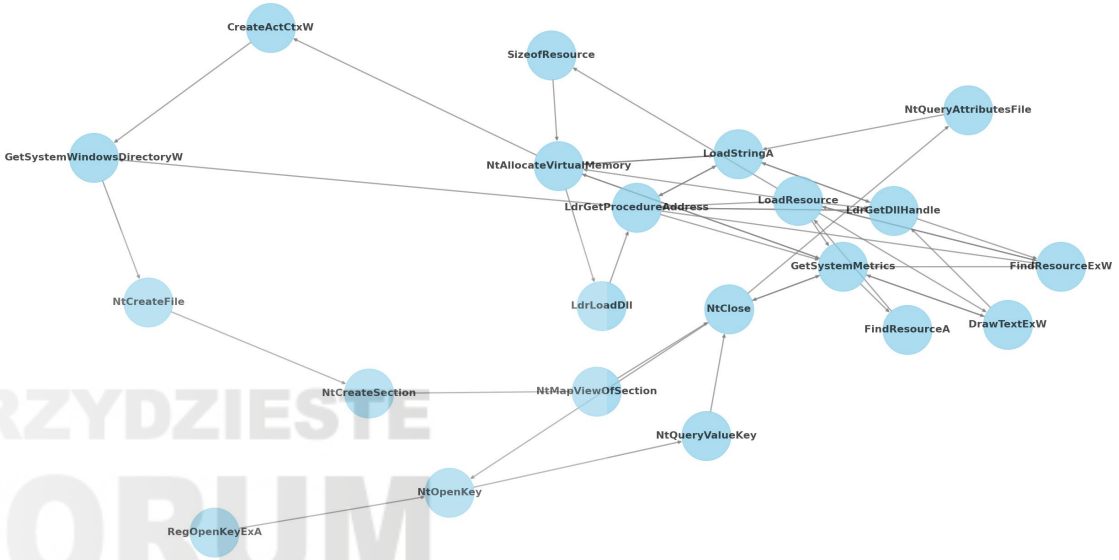
Podwójna architektura autoenkoderów SDA. SDA1 przyjmuje na wejście znormalizowaną poprzez node2vec reprezentację grafu CFG wywołań API, SDA2 natomiast na wejściu przyjmuje rzadki wektor zastosowań odpowiednich metod z API Windowsa w podejrzanym programie.

Źródło: „Jiang H., Turki T., Wang J., „DLGraph: Malware Detection Using Deep Learning and Graph Embedding," Proceedings of the 17th IEEE International Conference on Machine Learning and Applications (ICMLA), 17-20 Grudnia 2018, Orlando, USA, (2018)."



Budowa grafu zachowań bazująca na sekwencji wywołań funkcji API sytemu operacyjnego

Źródło: A. Oliveira i R. J. Sassi, „Behavioral Malware Detection Using Deep Graph Convolutional Neural Networks," International Journal of Computer Applications 174/29, 2021.



Przykładowy graf zachowań wirusa na system Windows zbudowany na podstawie danych uczących rozbudowywanego programu badawczego

Źródło: Opracowanie własne

Wykorzystanie lepszej normalizacji macierzy sąsiedztwa

$$S = D^{-\frac{1}{2}} \hat{A}\, D^{-\frac{1}{2}}$$

$$zamiast$$

$$S = D^{-1} A$$

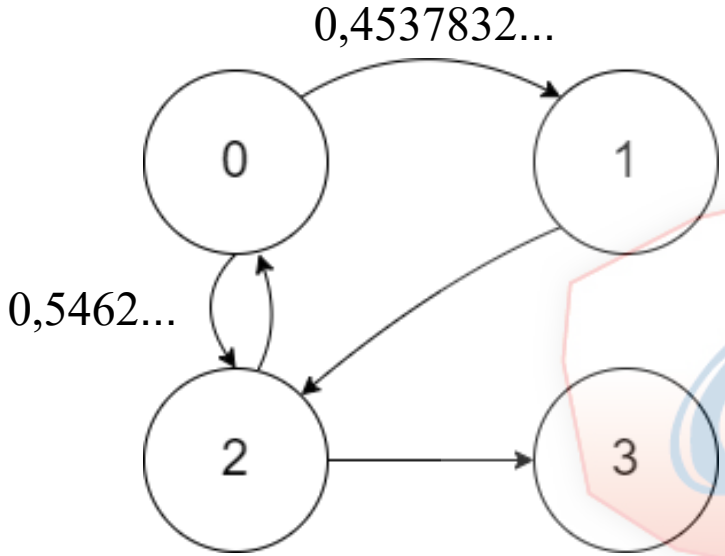
Gdzie:

$D^{-\frac{1}{2}}$: $Macierz\ stopni\ wierzchołków\ D\ podniesiona\ macierzowo\ do\ potęgi\ -\frac{1}{2}$

A: $Macierz\ sąsiedztwa$

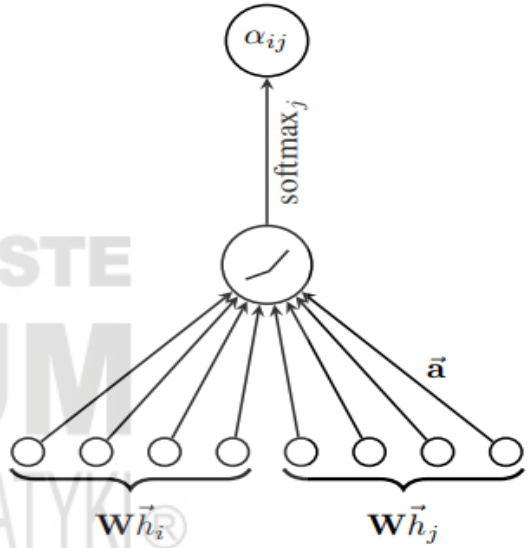
$\hat{A}$: Macierz sąsiedztwa A z pętlami

Zastosowanie mechanizmu uwagi (Graph Attention Network)



Przykładowy skierowany graf zachowań wykorzystywany w warstwie splotowej GCN. Prawdopodobieństwa transmisji pakietów obliczane są z wykorzystaniem uczenia perceptrona jednowarstwowego. Zapewnia to lepsze przypisanie struktury danych do analizowanego problemu.

Źródło rysunku: Opracowanie własne



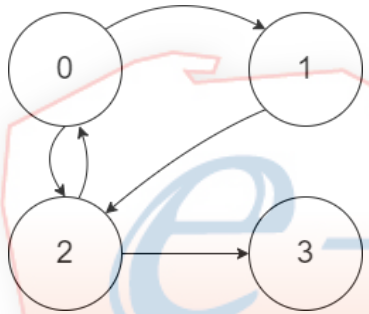
Perceptron jednowarstwowy stosowany do wyznaczania wag uwagi każdego z wierzchołków grafu wejściowego sieci GCN.

Źródło rysunku: P. Velicković i e. al., „GRAPH ATTENTION NETWORKS,” *ICLR*, Luty 2018.

Wykorzystanie architektury równoległych sieci GCN z lepszą normalizacją macierzy sąsiedztwa

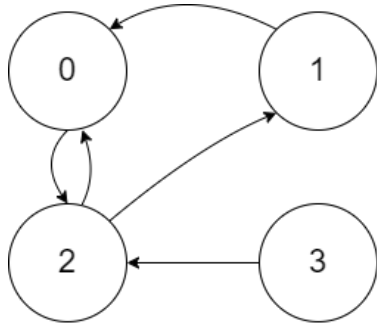
A

	0	1	2	3
0		1	1	
1			1	
2	1			1
3				



A^T

	0	1	2	3
0			1	
1	1			
2	1	1		
3			1	



Przykładowe grafy zachowań wykorzystywane w warstwie splotowej GCN z ulepszoną normalizacją macierzy sąsiedztwa A. Graf po prawej stronie ma przeciwnie skierowane łuki względem grafu po lewej. Zbudowanie bliźniaczych sieci neuronowych GCN w oparciu o obie reprezentacje jednocześnie zapewnia jeszcze lepsze odwzorowanie rzeczywistych danych w strukturę grafu. Sieć wykorzystująca graf oparty na A normalizuje macierz stopniami wyjścia wierzchołków, sieć A^T natomiast – normalizuje macierz sąsiedztwa stopniami wejścia wierzchołków. Wyniki uczenia obu sieci są agregowane w warstwie w pełni połączonej.

Źródło: opracowanie własne

Analiza porównawcza zaimplementowanych metod

	dokładność	precyzja	czułość	wskaźnik F1	ROC AUC
GCN	0,9275	0,9348	0,9205	0,9276	0,9275
GCN z lepszą normalizacją	0,9367	0,9414	0,9327	0,9370	0,9368
GCN z lepszą normalizacją, 2 warstwowa	0,9213	0,9110	0,9310	0,9209	0,9214
GCN z lepszą normalizacją, 3 warstwowa	0,9228	0,9113	0,9342	0,9226	0,9230
GCN z lepszą normalizacją, 4 warstwowa	0,9398	0,9321	0,9467	0,9393	0,9399
GCN z lepszą normalizacją, 5 warstwowa	0,9151	0,9681	0,8558	0,9085	0,9142
GCN z lepszą normalizacją, 6 warstwowa	0,9336	0,9510	0,9122	0,9312	0,9333
GCN z lepszą normalizacją, 7 warstwowa	0,9090	0,9140	0,8997	0,9068	0,9088
GCN z lepszą normalizacją, 8 warstwowa	0,8796	0,8453	0,9248	0,8832	0,8803
GCN z lepszą normalizacją, 9 warstwowa	0,8843	0,8526	0,9248	0,8872	0,8849
GCN z lepszą normalizacją, 10 warstwowa	0,8904	0,8804	0,8997	0,8899	0,8906
GAT	0,9059	0,9057	0,9028	0,9042	0,9058
GAT, 2 wymiarowa	0,8318	0,7793	0,9185	0,8432	0,8331
GCN równoległa	0,9306	0,9281	0,9310	0,9296	0,9306

Tabela przedstawia wartości zdefiniowanych miar jakości klasyfikatorów binarnych dla każdego ze zbudowanych modeli. W pierwszym wierszu tabeli zaprezentowane są wyniki reprodukcji oryginalnego algorytmu autorów badanego artykułu (autorzy w publikacji podają, że ich model uzyskał jeszcze lepsze wartości parametrów, jednakże zdecydowano, że punktem odniesienia w tym badaniu będą najwyższe zreprodukowane wyniki, jakie udało się uzyskać lokalnie autorowi pracy).
W powyższym zestawieniu czterowarstwowa sieć splotowa GCN uzyskała najwyższe wartości niemalże wszystkich parametrów – model ten uznano za najskuteczniejszy z zaimplementowanych.

Źródło: opracowanie własne

Perspektywy rozwoju

- Rozbudowanie zbioru danych uczących o wektory cech
- Wykorzystanie potencjału sieci równoległych
- Zastosowanie struktur grafowych dla innych zastosowań cyberbezpieczeństwa