SAFE: Self Attentive Function Embedding for binary similarity

16th Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA 2019)

Luca Massarelli\textsuperscript{1}, Giuseppe Antonio Di Luna\textsuperscript{2}, Fabio Petroni\textsuperscript{3}, Roberto Baldoni\textsuperscript{1}, Leonardo Querzoni\textsuperscript{1},

\textsuperscript{1}University of Rome ”La Sapienza”

\textsuperscript{2}CINI, National Laboratory of Cyber Security

\textsuperscript{3}Facebook AI Research

Gothenburg, June 20, 2019
Intelligent devices enable new "smart" production processes. More and more organizations rely on them every day. Organizations do not develop their own devices, but mostly rely on commercial ones.
COTS\(^a\) devices are provided as Black-Box with no access to their firmware's source code. While improving production processes, organizations have to trust devices manufacturers for assessing the absence of vulnerabilities or backdoors.

\(^a\)Commercial Off-the-Shelf
“Trust is good, control is even better.”

Even for a COTS device it is still possible to analyze its binary firmware, but this process is time consuming and requires skilled personnel.

There is a strong need of new tools that enable more efficient analysis of binary code. Natural Language Processing (NLP) techniques has proved to be powerful when applied to binary code.
A common approach in NLP is to associate to an entity (e.g. a word, a sentence, a whole text ...) an embedding vector, i.e. a fixed size vector of real numbers that contains information on the entity it represents.

Defining a relation between entities we can build a model that can be able to represent entities with embedding preserving the chosen relation.

"binary" $\mapsto \begin{bmatrix} 0.846 & 0.332 & \ldots & 0.954 \end{bmatrix}$

"binaries" $\mapsto \begin{bmatrix} 0.844 & 0.334 & \ldots & 0.984 \end{bmatrix}$
**Similarity Definition**

**Definition: Similar Functions**

Two binary functions are considered similar if they have been compiled from the same source code but possibly using different compilers, different optimizations and/or for different platforms.
Similarity Definition

Definition: Similar Functions

Two binary functions are considered similar if they have been compiled from the same source code but possibly using different compilers, different optimizations and/or for different platforms.

How to compute similarity-preserving embeddings for a binary function?
Related Work

No-Embeddings

- Bindiff [STICC-05]
- Strand [PLDI-16]

Embeddings

- Genius [CCS-16]
- Cross platform
**GENIUS** by ¹ showed that a binary function can be represented with a similarity-preserving vector. That is, given two similar functions, their embedding vectors should be similar in terms of cosine similarity.

- Computing the cosine similarity of two vectors is extremely faster than comparing two graphs.
- The Binary-Similarity problem has been reduced to the computation of similarity-preserving function embeddings.

---
Related Work

- Genius [CCS-16]
- Gemini [CCS-17]
- Strand [PLDI-16]
- Embeddings
- No-Embeddings

Cross platform


Strand [PLDI-16] → Genius [CCS-16]
**Gemini** by Xiaojun et al.\(^2\) proposes a graph embedding deep neural network\(^3\) to produce an embedding vector of the annotated control flow graph (ACFG) of a function.


Related Work

No-Embeddings

Bindiff [STICC-05]

Strand [PLDI-16]

Genius [CCS-16]

Gemini [CCS-17]

Unsupervised feature learning [BAR-19]

Embeddings

Inner eye [NDSS-18] Solves a subproblem

Stripped Binaries

Single platform

Asm2Vec [SP-19]

Cross platform
Massarelli et al. proposed a modified version of Gemini where node’s features are learned during training stage.

\[ f = (3.12, \ldots, 5.31) \]

---

\[ L. \text{Massarelli et al. Investigating Graph Embedding Neural Networks with Unsupervised Features Extraction for Binary Analysis, BAR 19} \]
Motivations of our work

Exploring efficient way to compute function embeddings

Determine if embeddings can be used to afford other tasks
Motivations of our work

Exploring efficient way to compute function embeddings

Determine if embeddings can be used to afford other tasks

Tasks:

- **Binary Similarity**: Given two binary functions detect if they have been compiled from the same source code.

- **Function Retrieval**: Given a target function retrieve all similar functions from a knowledge base.

- **Semantic Classification**: Given a binary function identify the semantic class of the function.
Each instruction is associated with an embedding vector using the \( i2v \) layer;

Instructions are fed into a \textit{Self-Attentive RNN};
Attention model are well studied by NLP community\(^5\). Since they perform very well on different NLP tasks we try to use attention to analyze binary code.

Datasets

- **AMD64ARMOpenSSL Dataset**: 95535 functions, Two versions of OpenSSL\(^6\) compiled for ARM and X86 using gcc.
- **AMD64multipleCompilers Dataset**: 452598 functions, Nine different open source projects\(^7\) compiled for X86 platform.
- **AMD64PostgreSQL Dataset**: 581640 functions, PostgreSQL 9.6.0 compiled with 12 different compilers\(^8\) for X86 platform.
- **Semantic Dataset**: 15158 functions, implementations of different 443 c functions that have been manually annotated as implementing algorithms in one of the 4 classes: Encryption, Sorting, String Manipulation, Mathematical.

---

\(^6\) v1.0.1f - v1.0.1u

\(^7\) binutils-2.30, ccv0.7, coreutils-8.29, curl- 7.61.0, gsl-2.5, libhttpd-2.0, openmpi-3.1.1, openssl- 1.1.1-pre8, valgrind-3.13.0

\(^8\) gcc- 3.4, gcc-4.7, gcc-4.8, gcc-4.9, gcc-5.4, gcc-6, gcc-7, clang-3.8, clang-3.9, clang-4.0, clang-5.0, clang-6.0
Task 1: Binary Similarity

Evaluation of embedding quality using the Receiver Operating Characteristic (ROC) curve on the test set.

**AMD64ARMOpenSSL Dataset.**
SAFE AUC: 0.99, GEMINI AUC: 0.95

**AMD64multipleCompilers Dataset.**
SAFE AUC: 0.99, GEMINI AUC: 0.93
Task 2: Functions Retrieval

We test SAFE single platform trained model as a function search engine:

- Given the embedding of a query function we want to retrieve all functions similar from a knowledge base.

Evaluation metrics:

- Precision @ $k$: Precision over the first $k$ retrieved functions.
- Recall @ $k$: Recall over the first $k$ retrieved functions.
- Normalized Discounted Cumulative Gain:

$$nDCG(R_f) = \sum_{i=1}^{k} \frac{isSimilar(r_i, \tilde{f})}{log(1+i)}$$

$$\text{IdealDCG}_k$$
Task 2: Functions Retrieval

Varying $k$ we evaluated the three metrics over the AMD64PostgreSQL Dataset.

<table>
<thead>
<tr>
<th></th>
<th>SAFE</th>
<th>GEMINI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision @ 10</td>
<td>84%</td>
<td>74%</td>
</tr>
<tr>
<td>Recall @ 10</td>
<td>21%</td>
<td>18%</td>
</tr>
<tr>
<td>nDCG @ 10</td>
<td>85%</td>
<td>75%</td>
</tr>
</tbody>
</table>
Task 3: Functions Semantic Classification

Using SAFE and GEMINI embeddings we try to classify the semantic of the functions in the Semantic Dataset. Embeddings are classified using a linear SVM in four semantic classes.

Confusion matrix with GEMINI embeddings

Accuracy: 0.89

Confusion matrix with SAFE embeddings

Accuracy: 0.95
Task 3: Functions Semantic Classification

Projecting SAFE embeddings of Semantic Dataset functions it is possible to identify also semantic clusters.
Main Takeaways

We show that using a Recurrent Neural Network with self attention permits to compute function embeddings with several advantages:

1. **Precision**, since SAFE embedding capture better similarity between functions;

2. **Efficiency**, since it is not required the computation of the CFG;

3. **Semantic Similarity**, since it is possible to capture the semantic similarity between functions;
We plan to continue this research following two different strategies:

- **Increment SAFE performances:** Recent works like Asm2Vec\(^9\) show that information on the CFG can be taken into accounts considering paths on the graph.

- **Extensive study on semantic classification:** We want to deeply study this problem defining a larger dataset and evaluating new training strategies and new model to tackle this tasks.

---

https://github.com/gadiluna/SAFE
https://github.com/gadiluna/SAFE

Thanks for your attention!

Mail: massarelli@diag.uniroma1.it