Taiyan WANG, Qingsong XIE, Lu YU, Zulie PAN, Min ZHANG, 2025. A survey of binary code representation technology. *Frontiers of Information Technology & Electronic Engineering*, 26(5):671-694. https://doi.org/10.1631/FITEE.2400088

A survey of binary code representation technology

Key words: Binary analysis; Binary code representation; Binary code feature selection; Binary code feature embedding

Corresponding author: Min ZHANG
E-mail: zhangmindy@nudt.edu.cn
ORCID: https://orcid.org/0000-0002-6654-7610

Binary code representation

- Binary code representation refers to extracting the features of the binary program and creating an embedded representation in the numerical vector space based on these features.
- The overall binary code representation learning process can be divided into two steps: (1) binary code feature selection; (2) binary code feature embedding.

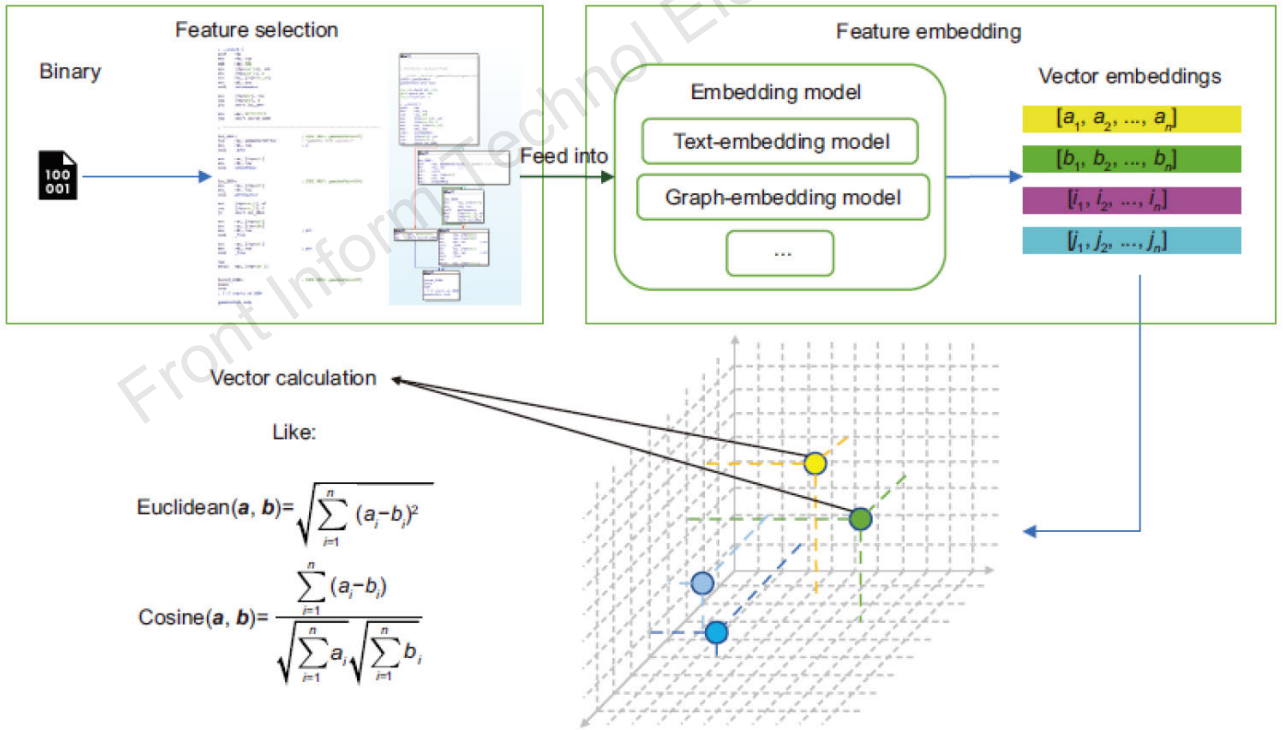

Fig. 2 Binary code representation procedure

Binary code representation

- The downstream tasks in binary analysis comprise a range of specific undertakings that leverage binary code analysis technology.
- These encompass tasks such as variable type and name prediction, function information recovery, binary code similarity detection, and malicious code detection, among others.

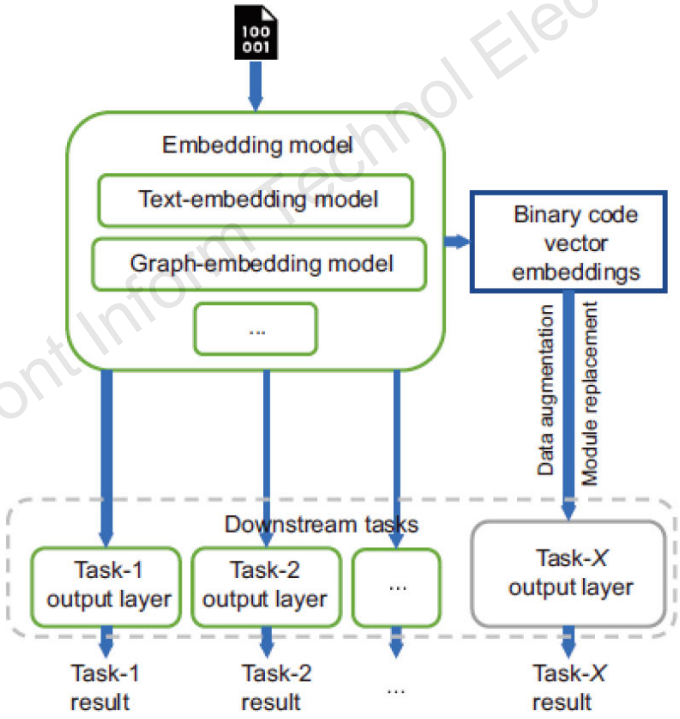

Fig. 3 Workflow of using binary code representation in binary analysis downstream tasks

Binary code feature selection

- The crux of binary code representation lies in selecting its semantic features, which involves:
  - **Binary code feature classification**
    - Nine different categories are defined;
    - Different approaches consider one or several of these for their investigations.
  - **Binary code feature construction**
    - Six types are defined;
    - Each one is the combination of one or more of the above nine features.

Binary code feature selection

Table 3 Selection of code features in existing research works

Name	Binary code feature category									Form of characterization
	Syntax feature	Textual feature	Statistical feature	Control flow feature	Data flow feature	Symbolic feature	Function call feature	Data resource	Dynamic feature	
Genius (Feng et al., 2016)			✓	✓						ACFG
Gemini (Xu XJ et al., 2017)			✓	✓						ACFG
αDiff (Liu BC et al., 2018)							✓			Raw bytes
VulSeeker (Gao J et al., 2018a)			✓	✓	✓					LSFG
VulSeeker-Pro (Gao J et al., 2018b)			✓	✓	✓				✓	LSFG, dynamic trace
Zeek (Shalev and Partush, 2018)		✓		✓	✓					Strands
SAFE (Massarelli et al., 2019b)		✓								Assembly
GMN (Li YJ et al., 2019)				✓						ACFG
InnerEye (Zuo et al., 2019)		✓								Assembly
cross-arch-instr-model (Redmond et al., 2019)		✓								Assembly
GraphEmb (Massarelli et al., 2019a)		✓		✓						ACFG
Asm2Vec (Ding et al., 2019)		✓								Assembly
Order Matters (Yu ZP et al., 2020b)		✓		✓						ACFG, CFG
PatchEcko (Sun et al., 2020)			✓						✓	Numerical statistical feature
MKIS (Li YC et al., 2020)		✓		✓					✓	Customized instruction sequence
DeepBinDiff (Duan et al., 2020)		✓		✓						ACFG
MIRROR (Zhang XC et al., 2020)		✓								Assembly
BCSD (Liu ZA, 2021)				✓		✓				Customized graph structure
PalmTree (Li XZX et al., 2021)		✓			✓					Assembly
Asteria (Yang SG et al., 2021)	✓						✓			AST
OSCAR (Peng et al., 2021)		✓								IL text
$\text{BinDiff}_{\text{NN}}$ (Ullah and Oh, 2022)		✓								Assembly
Codee (Yang J et al., 2022)		✓		✓						Assembly, ACFG
BinSeeker (Gao J et al., 2021)			✓	✓	✓				✓	LSFG, dynamic trace
BinShot (Ahn et al., 2022)		✓								Assembly
BMM (Guo YX et al., 2022)		✓		✓	✓					CFG, call graph, DFG
jTrans (Wang H et al., 2022)		✓		✓						Customized instruction sequence
XBA (Kim G et al., 2022)		✓		✓			✓	✓		Customized graph structure
Trex (Pei et al., 2023)		✓			✓				✓	Customized instruction sequence
TikNib (Kim D et al., 2023)			✓							Numerical statistical features
DiEmph (Xu XZ et al., 2023)		✓								Customized instruction sequence
VulHawk (Luo et al., 2023)		✓		✓			✓	✓		ACFG
Asteria-Pro (Yang SG et al., 2023)	✓						✓	✓		AST
sem2vec (Wang HJ et al., 2023b)		✓		✓		✓	✓		✓	Dynamic trace, function call sequence
BinFinder (Qasem et al., 2023)		✓					✓	✓		IL text, function call sequence, constants

Binary code feature embedding

- This section introduces the feature embedding representation component of existing binary code representation methods, focusing on:

- Methods using text-embedding models
- Methods using graph-embedding models
- Methods integrating text-embedding and graph-embedding models
- Other methods

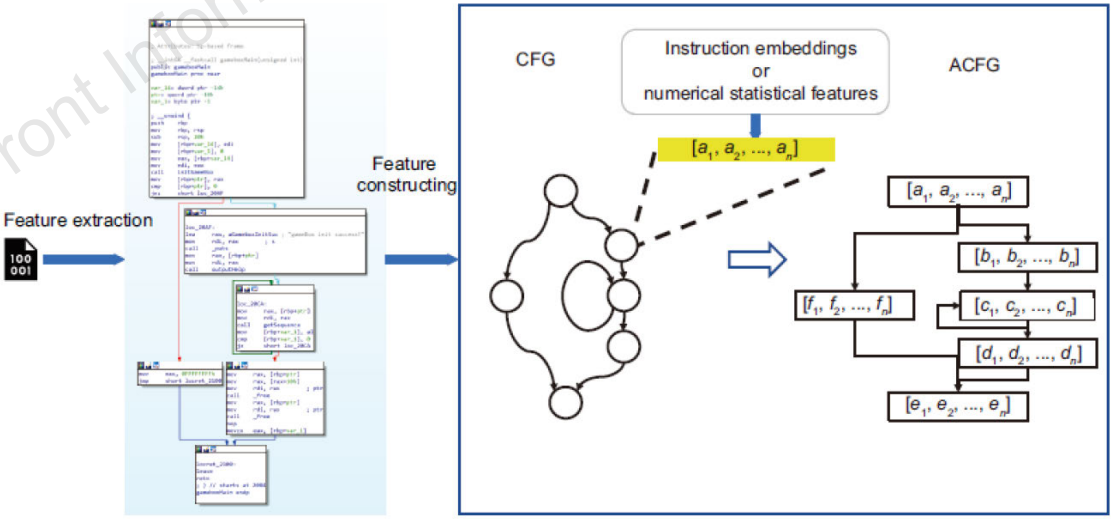

Fig. 4 ACFG construction process

Binary code feature embedding

Table 4 Embedding methods of code features in existing research works

Method	Feature-embedding method		
	Text-embedding model	Graph-embedding model	Others
Genius (Feng et al., 2016)			Spectral clustering
Gemini (Xu XJ et al., 2017)		structure2vec	
αDiff (Liu BC et al., 2018)		CNN	
VulSeeker (Gao J et al., 2018a)		Customized DNN	
VulSeeker-Pro (Gao J et al., 2018b)		Customized DNN	
Zeek (Shalev and Partush, 2018)			Customized layers
SAFE (Massarelli et al., 2019b)	Word2Vec (Skip-Gram)		
GMN (Li YJ et al., 2019)		Graph matching network	
InnerEye (Zuo et al., 2019)	Word2Vec (Skip-Gram)		
cross-arch-instr-model (Redmond et al., 2019)	Word2Vec (CBOW)		
GraphEmb (Massarelli et al., 2019a)	Word2Vec (Skip-Gram)	structure2vec	
Asm2Vec (Ding et al., 2019)	PV-DM		
Order Matters (Yu ZP et al., 2020b)	BERT	CNN+MPNN	MLP
PatchEcko (Sun et al., 2020)			Customized layers
DeepBinDiff (Duan et al., 2020)	Word2Vec (CBOW)	TADW	
MIRROR (Zhang XC et al., 2020)	Transformer		
PalmTree (Li XZX et al., 2021)	BERT		
Asteria (Yang SG et al., 2021)		Tree-LSTM	
OSCAR (Peng et al., 2021)	Transformer		
$\text{BinDiff}_{\text{NN}}$ (Ullah and Oh, 2022)	ABENN		
Codee (Yang J et al., 2022)	Word2Vec (Skip-Gram)	AANE+LINE	
BinSeeker (Gao J et al., 2021)		Customized DNN	
BinShot (Ahn et al., 2022)	BERT		
BMM (Guo YX et al., 2022)	BERT	GGNN	
jTrans (Wang H et al., 2022)	BERT		
XBA (Kim G et al., 2022)		GCN	
Trex (Pei et al., 2023)	Transformer		
DiEmph (Xu XZ et al., 2023)	Transformer		
VulHawk (Luo et al., 2023)	RoBERTa	GCN	
Asteria-Pro (Yang SG et al., 2023)		Tree-LSTM	
sem2vec (Wang HJ et al., 2023b)	RoBERTa	GCN	
BinFinder (Qasem et al., 2023)			Customized layers

Future prospects

- Multi-modal and large model-based representation
  - By leveraging this, we can generate more comprehensive representation vectors that encapsulate rich feature information.
- Source code and annotation enhanced representation
  - This approach allows for the description of binary code from two distinct levels: functional characteristics and design ideas.
- Interpretable binary code representation
  - Addressing this is integral to enhancing the overall comprehensibility and effectiveness.
- Downstream tasks of binary analysis
  - Numerous software engineering studies have resorted to AI techniques, particularly representation learning, as viable solutions.

Conclusions

- This paper provides a comprehensive survey of recent advances in binary code representation technology. First, we introduce the concept of binary code representation and discuss its correlation with downstream tasks in binary analysis. Subsequently, we categorize existing research workflow into two fundamental components: binary code feature selection methods and binary code feature embedding methods. Finally, we summarize the overall development of existing research and provide prospects for some potential research directions.