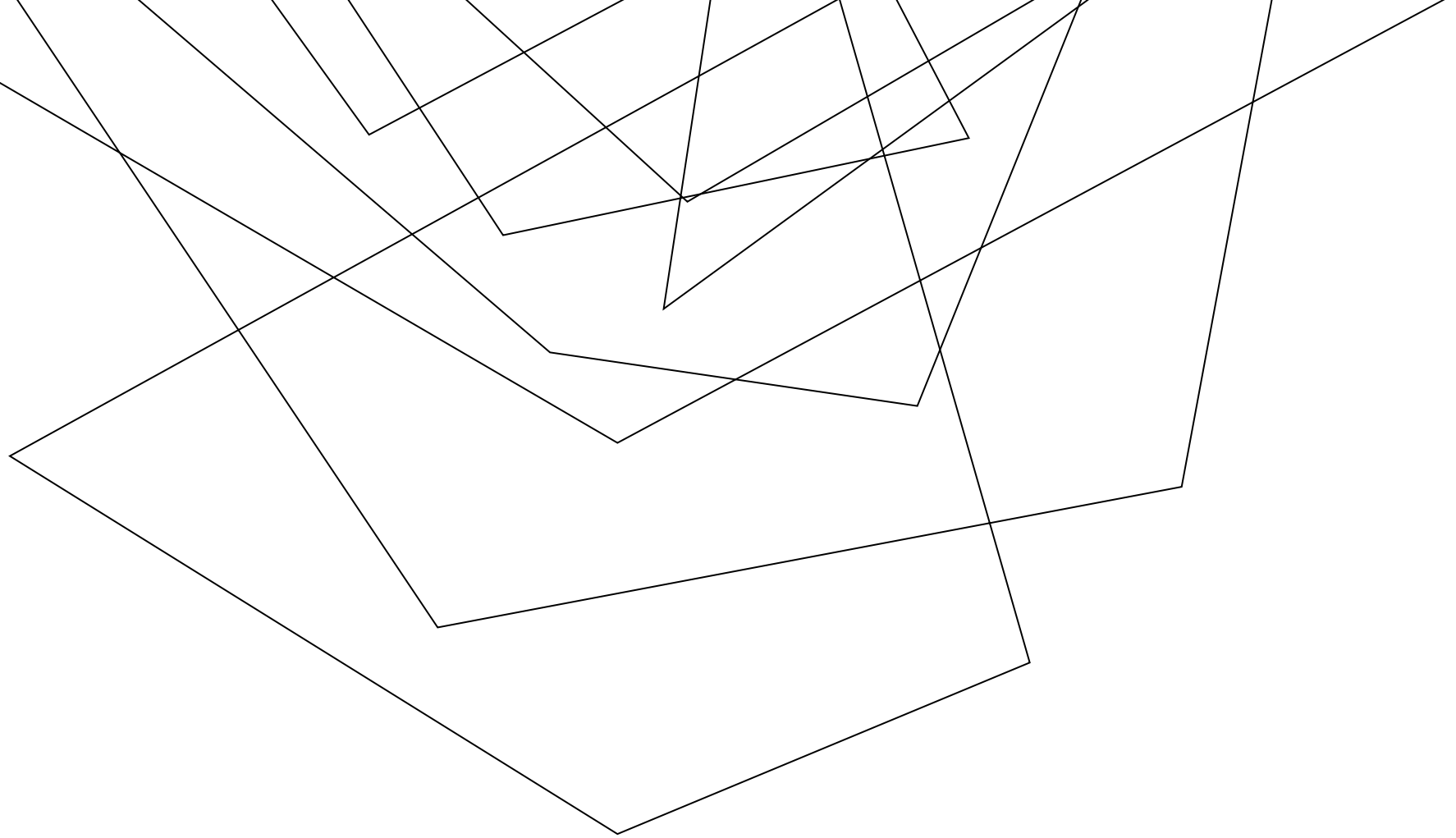
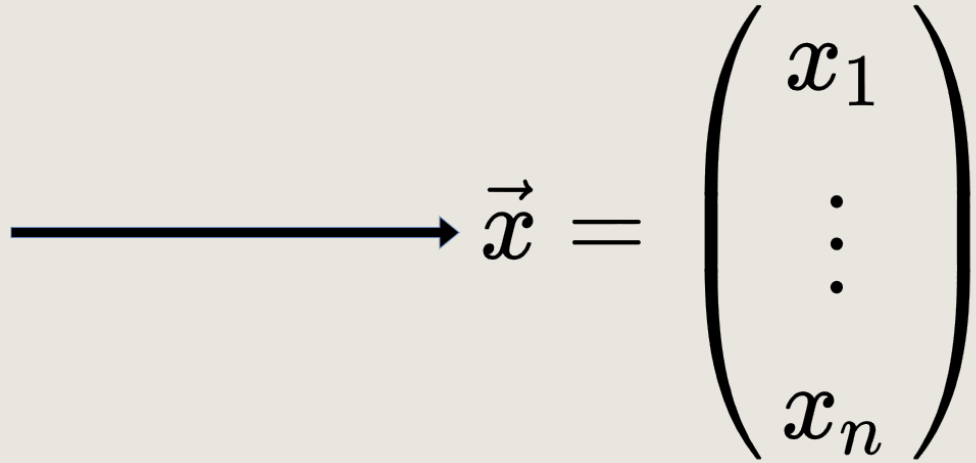


PITCH VORTRAG

Ruben Triwari

ASSEMBLY CODE EMBEDDING

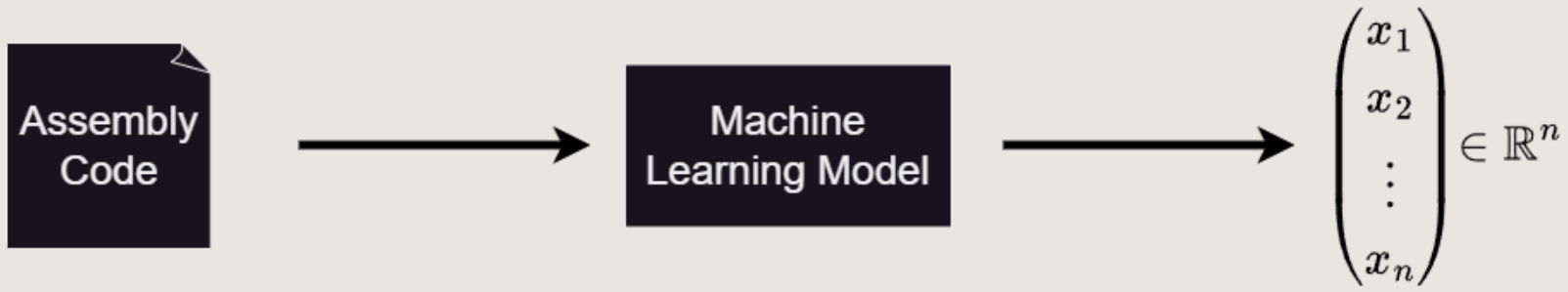
```
square(int):
        push    rbp
        mov     rbp, rsp
        mov     DWORD PTR [rbp-4], edi
        mov     eax, DWORD PTR [rbp-4]
        imul    eax, eax
        pop     rbp
        ret
```



- Nearest Neighbor Search
- Classification with Labels (e.g. Networking, time, ...)
- XFL: eXtreme Function Labeling
- Binary Code Similarity Detection (BCSD)

⇝ Assist Reverse Engineering

GENERATING ASSEMBLY EMBEDDINGS



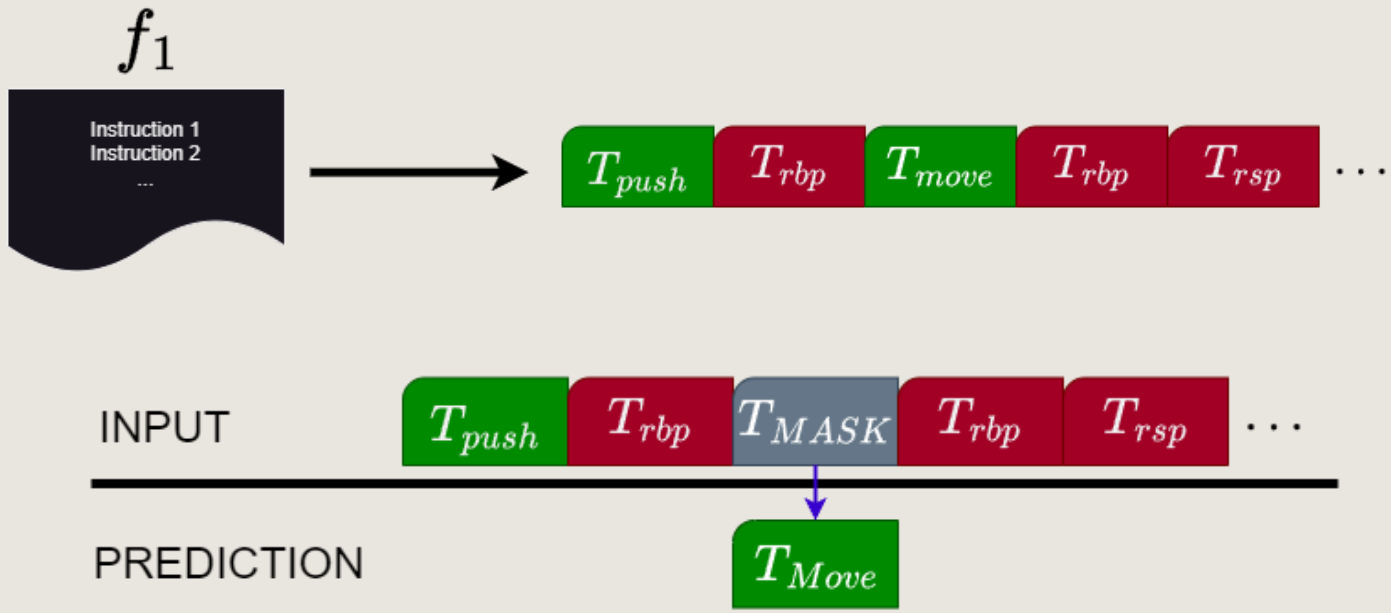
- Machine Learning Models to generate Assembly Embeddings
- Models need an objective function
  - ⇝ What is a “good” Objective function?

OVERVIEW OF DIFFERENT APPROACHES

- PalmTree[1] & JTrans[2]: BERT[3] like model to learn embeddings
- SAFE[4]: Supervised Learning with same Source as Label

[1]: PalmTree: X. Li, Q. Yu, H. Yin [2]: JTrans H. Wang, W. Qu, G. Katz, W. Zhu, Z. Gao, H. Qiu, J. Zhuge, C. Zhang
[3]: BERT J. Devlin, M.-W. Chang, K. Lee, K. Toutanova
[4] SAFE: L. Massarelli, G. Antonio D. Luna, F. Petroni, L. Querzoni, R. Baldoni

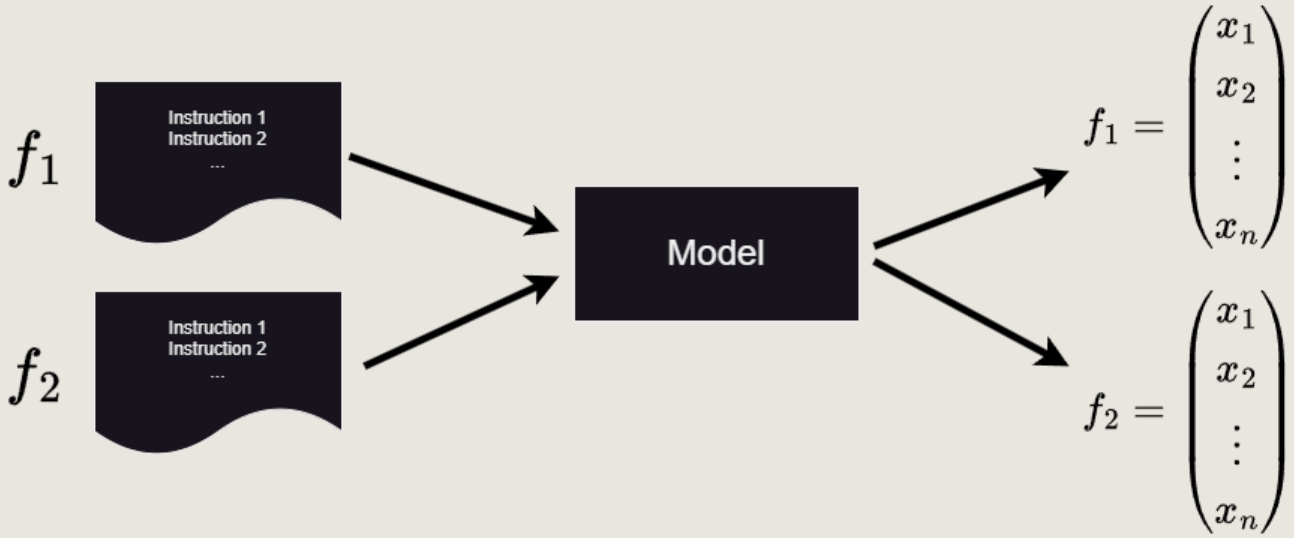
PALMTREE & JTRANS



- Self-supervised learning
- Tasks to train Model (e.g., Masking MLM)

  ⇝ State of the art in Binary Code Similarity Detection

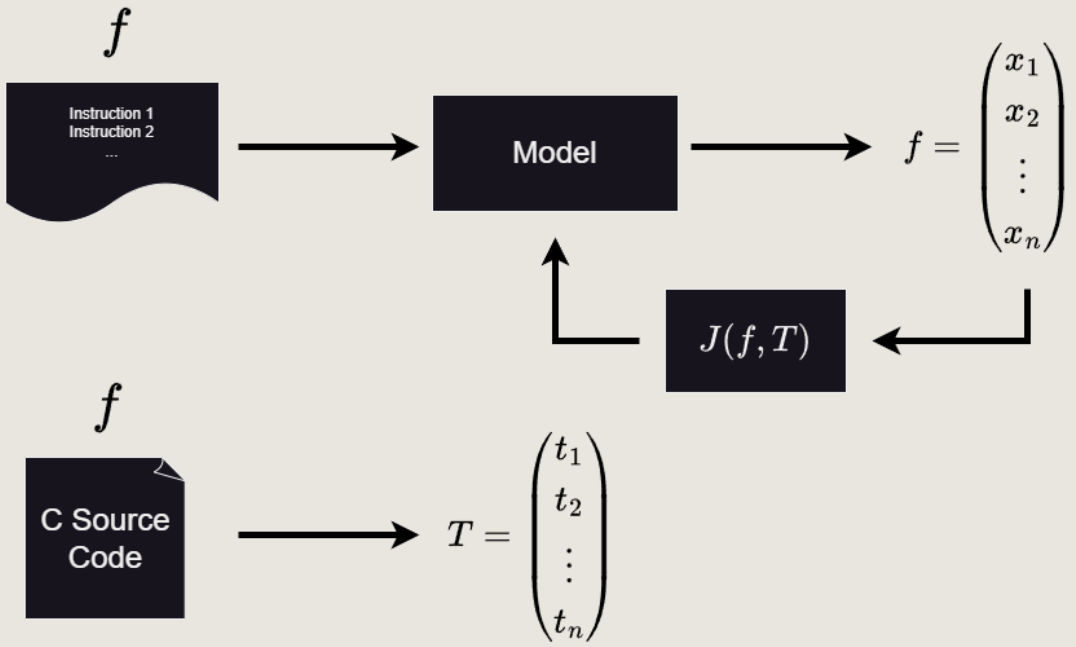
SAFE: SELF-ATTENTIVE FUNCTION EMBEDDINGS



- f_1 and f_2 same source code ⇝ minimize distance
- f_1 and f_2 different source code ⇝ maximize distance

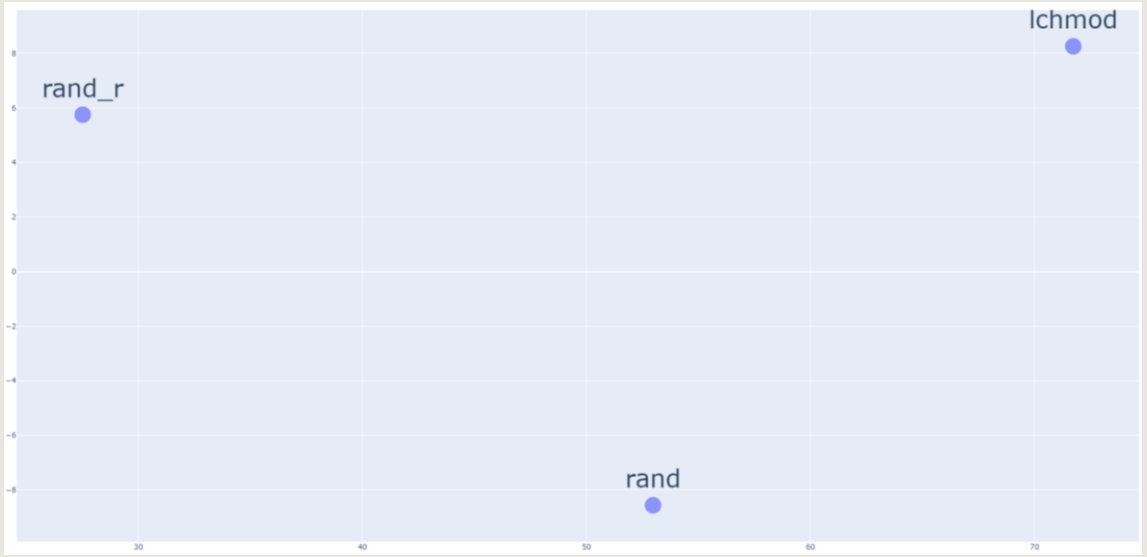
  ⇝ Similar functions are potentially far away from each other

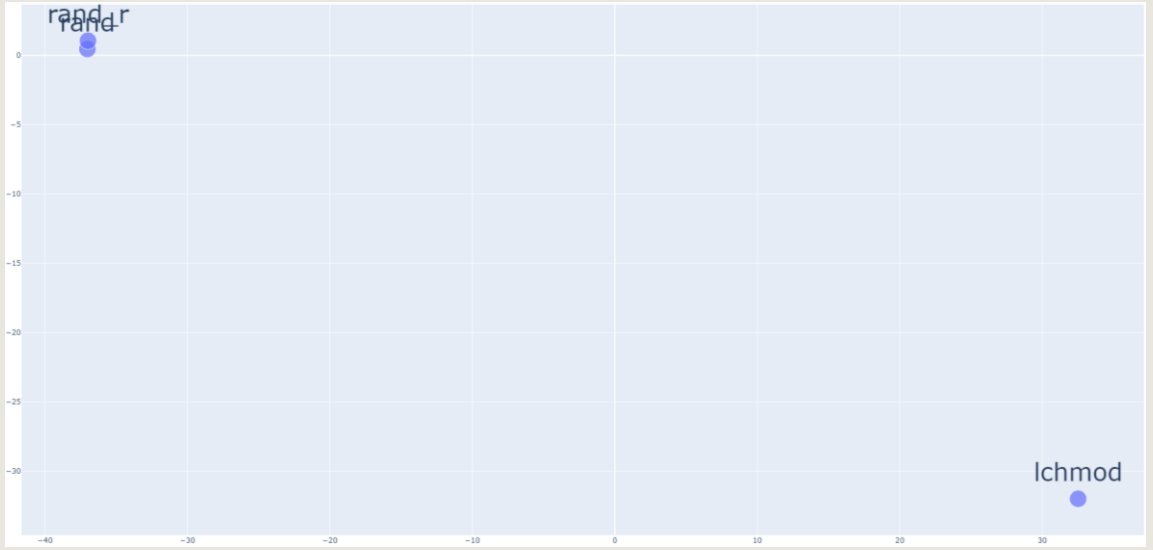
OUR APPROACH



- Supervised Learning
- Using C Source Code to generate “ground truth”

GROUND TRUTH





- Semantic similar functions ⇝ Vectors close to each other
- Semantic similar functions ⇝ Vectors far away from each other

  ⇝ Similar functions are grouped together

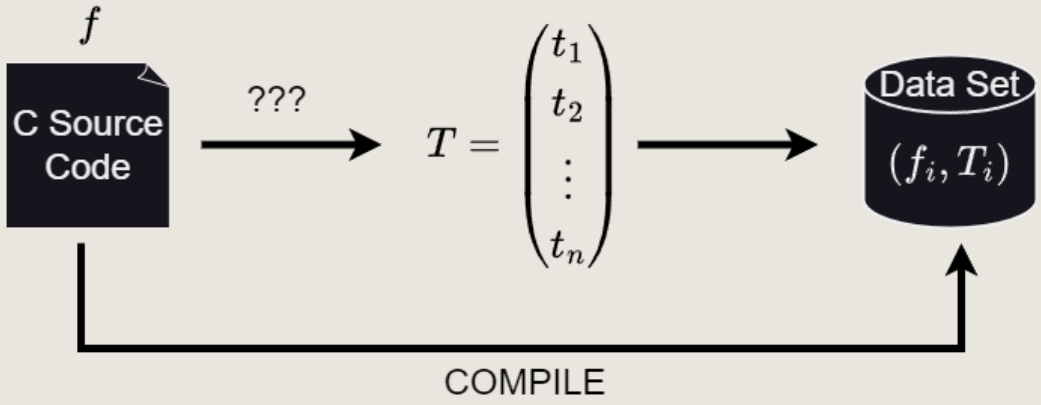
SENTENCE TRANSFORMER[1]

- Encoding Sentences semantically to Vectors
- Groups sentences with similar semantic

  ⇝ Exactly what we want for our ground truth

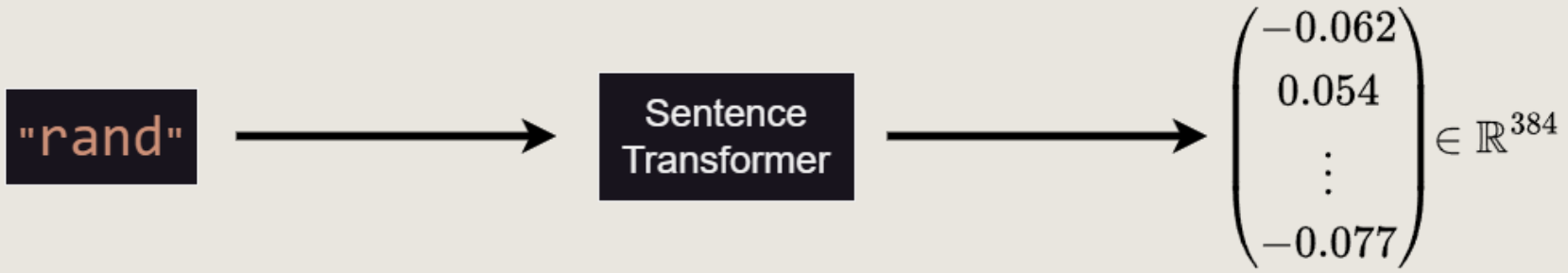
[1]: Sentence Transformer: https://www.sbert.net/

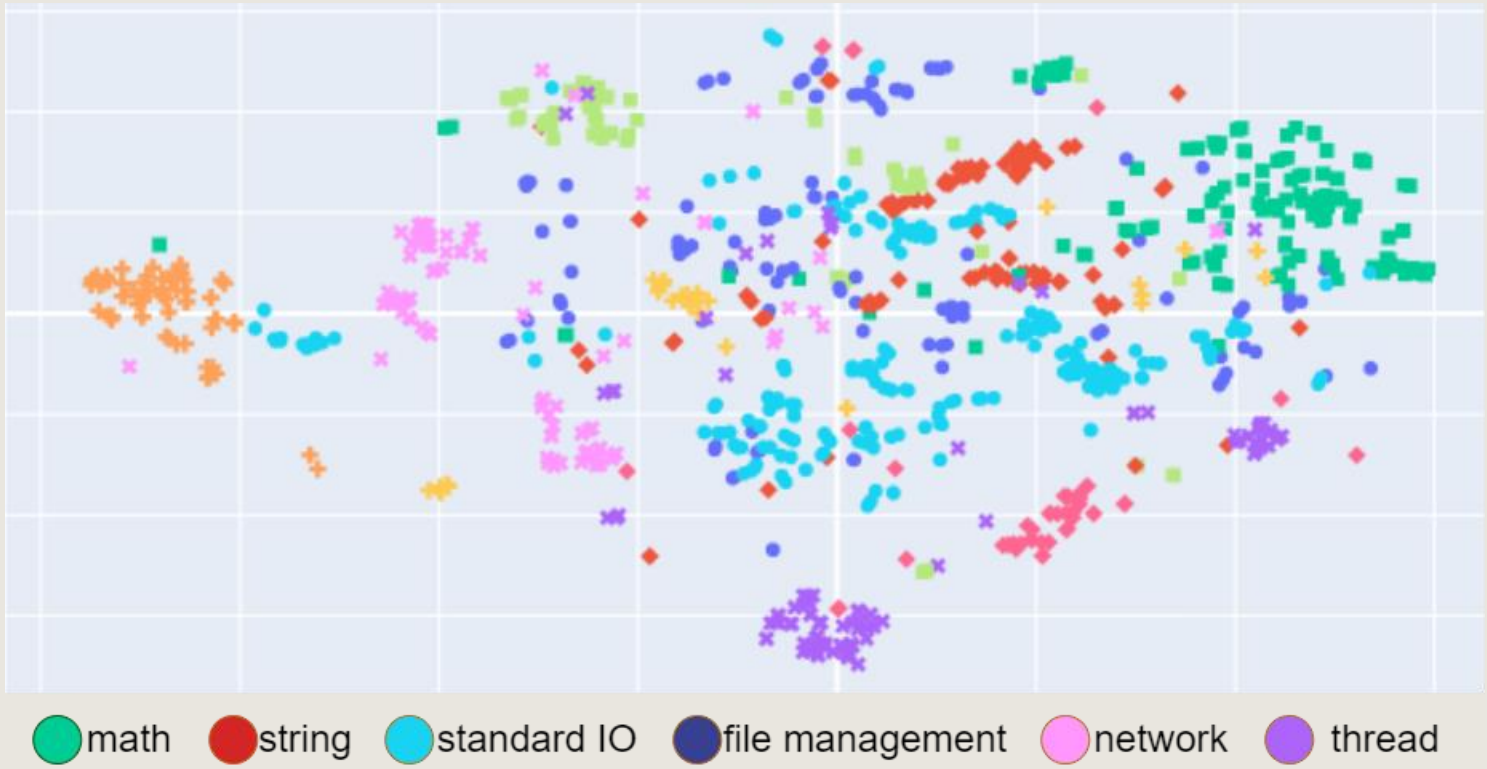
MY PART



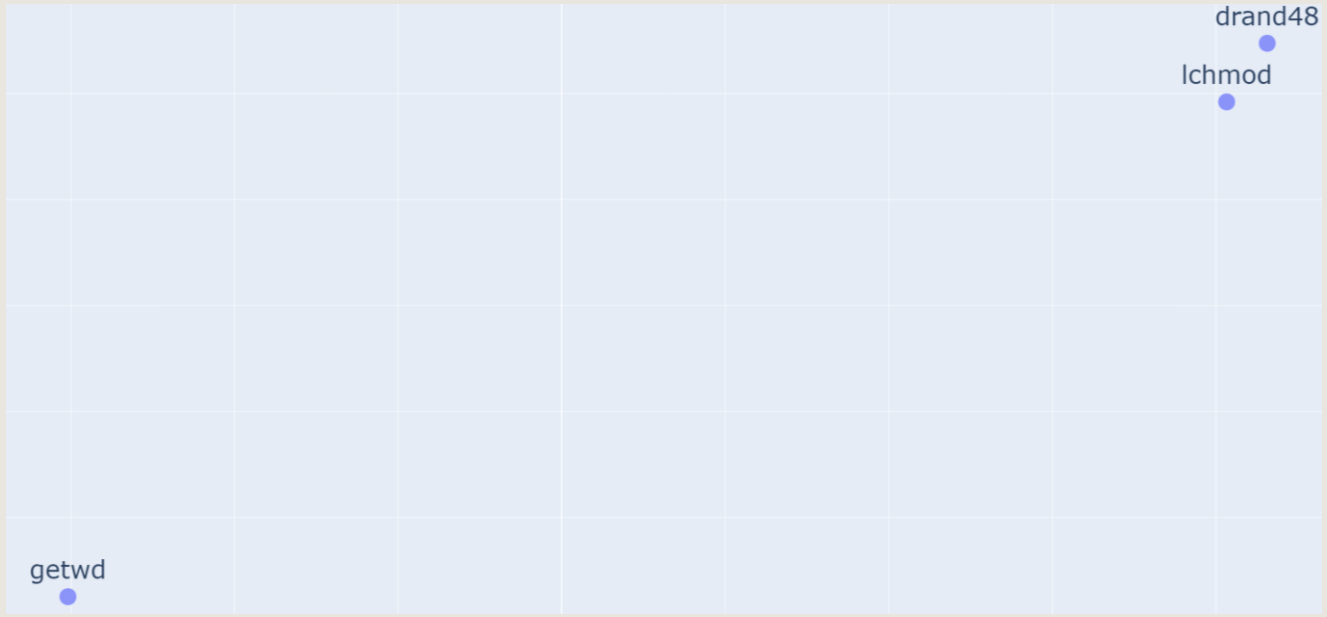
- Sentence Transformer on C Source Code Names
- Sentence Transformer on C Source Code Comments
- Sentence Transformer on LLM Code Summaries generated

C SOURCE CODE NAMES



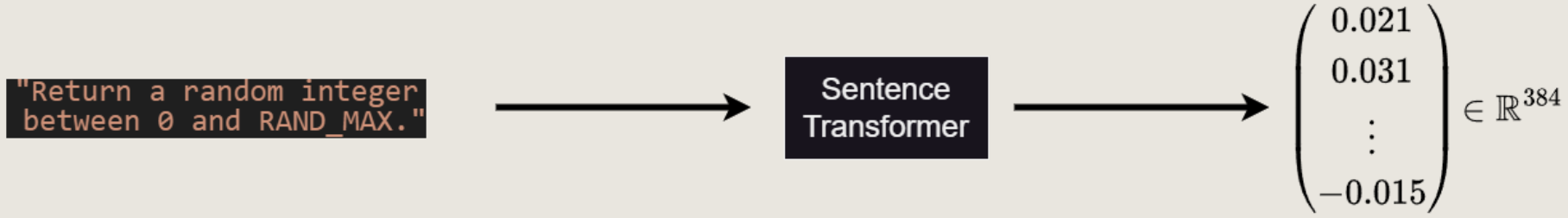


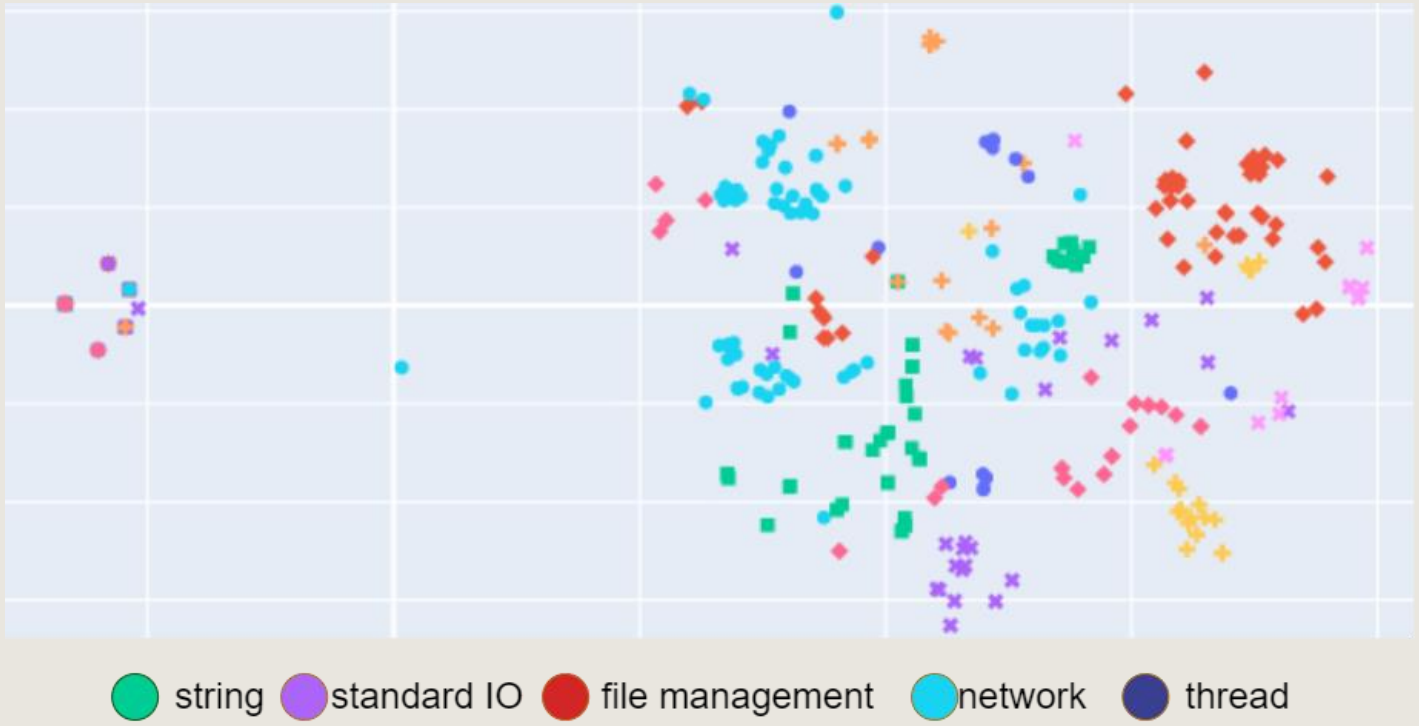
LIMITATION



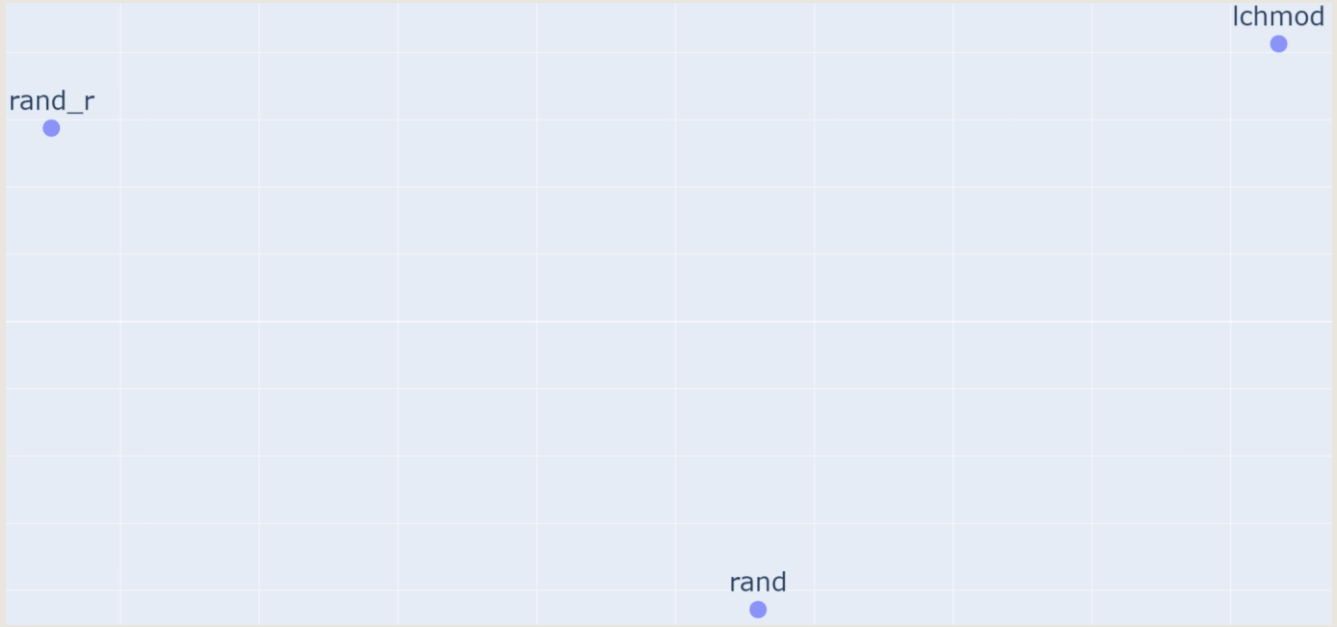
- “lchmod”, “getwd”: file operation
- “drand48”: Generates a random Number
  - ⇝ “lchmod” and “getwd” should be close
  - ⇝ Abbreviations can potentially confuse the sentence transformer

C SOURCE CODE COMMENTS





LIMITATION



- “rand_r” comment: “`This algorithm is mentioned in the ISO C…` “
- “rand” comment: “`Return a random integer between 0 and RAND_MAX.`”

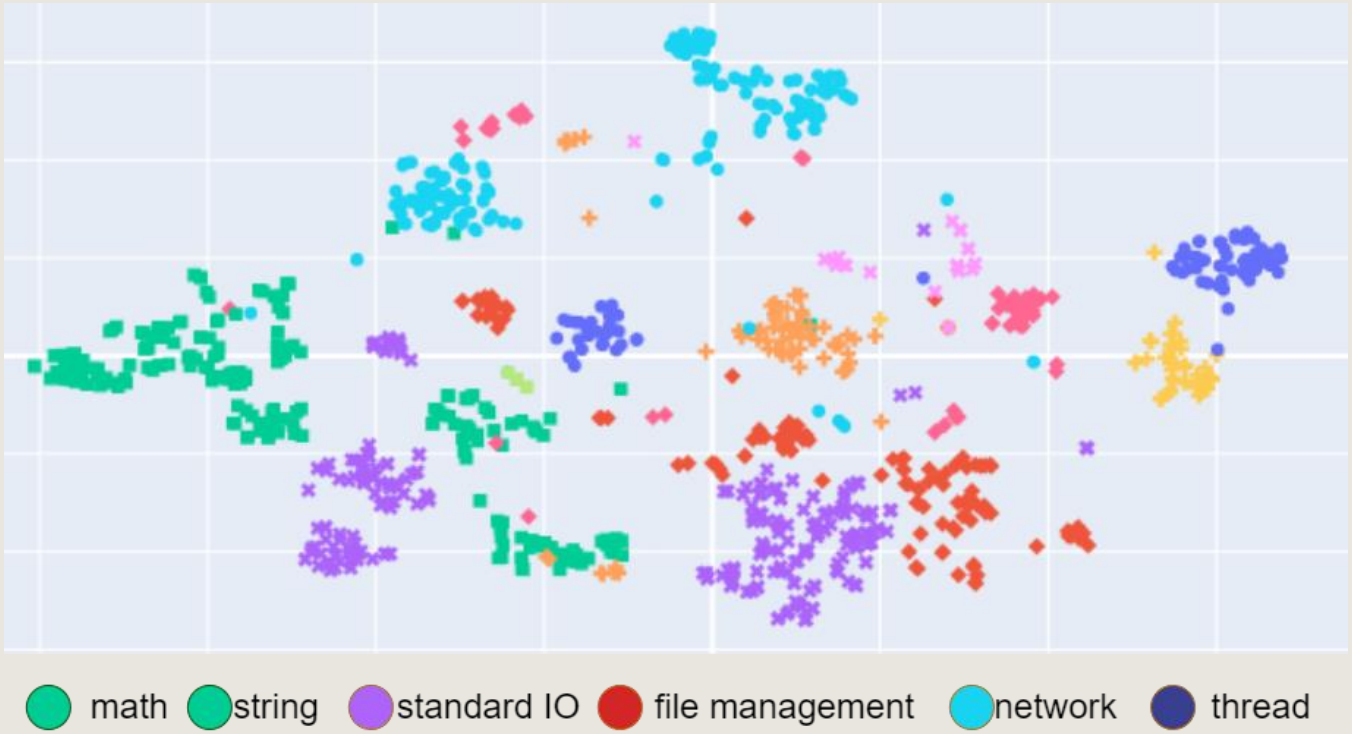
  ⇝ Comments are not Always summarizing the source code

  ⇝ Comments are not guaranteed to even exist

LLAMA

- Meta’s Large Language Model
- Open Source ⇝ runs on our server
- Prompt used: “Can you briefly summarize in one two sentence what the following function does? And can you give just the summary?\n”

LLAMA GENERATED CODE SUMMARIES



EVALUATION

- Qualitative evaluation:
  - Map Vectors in low dimension
  - Look for Cluster
- Quantitative evaluation:
  - Performance in downstream task

DISCUSSION

- What would be a better Prompt?
- C Embeddings with Code2Vec[1]
- Use Code Llama instead of standard Llama
- Alias problem in “Glibc” data set
- Other useful source code information I could extract?
- Combining vectors of different source code information?

[1]: Code2vec U. Alon, M. Zilberstein, O. Levy, E. Yahav