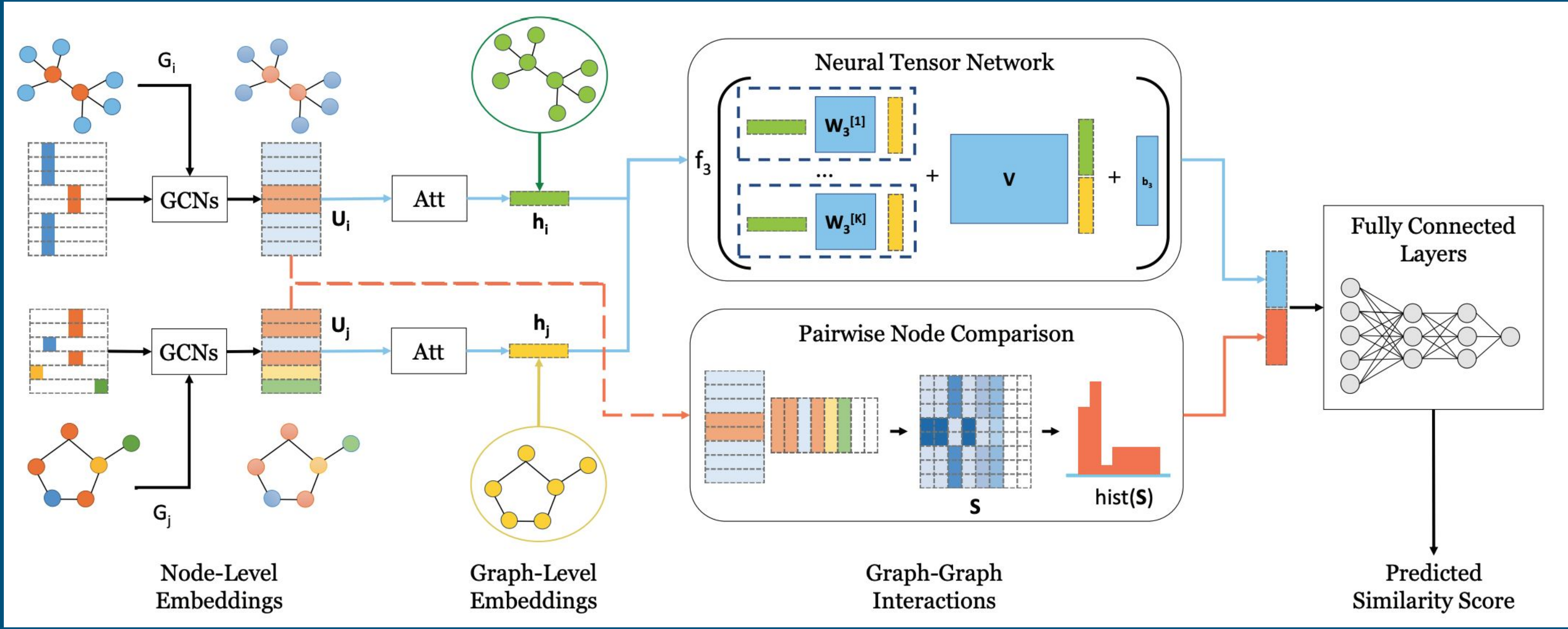
Survey of Graph Neural Networks in Programming Languages

- Sripath Mishra, Justin Yi, Hemil Desai

Classification and Similarity

SimGNN: Fast Graph Similarity Computation



SimGNN

- Graph Edit Distance (GED)
  - Computationally intensive to compute exactly
- In worst case, computation is quadratic in graph size, which is among SOTA for approximate graph distance computation
- Future Directions:
  - Integration of edge features
  - Scalability to large graphs

ρ - Spearman's Rank Correlation Coefficient

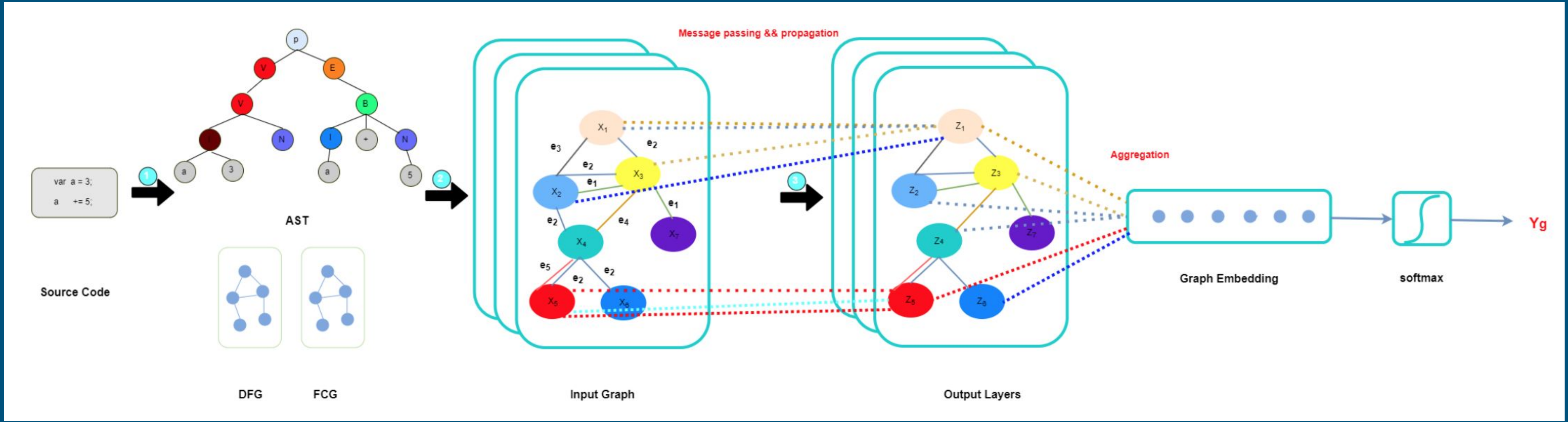
τ - Kendall's Rank Correlation Coefficient

p@k - Precision at k

Table 4: Results on LINUX.

Method	mse(10^{-3})	ρ	τ	p@10	p@20
Beam	9.268	0.827	0.714	**0.973**	0.924
Hungarian	29.805	0.638	0.517	0.913	0.836
VJ	63.863	0.581	0.450	0.287	0.251
SimpleMean	16.950	0.020	0.016	0.432	0.465
HierarchicalMean	6.431	0.430	0.525	0.750	0.618
HierarchicalMax	6.575	0.879	0.740	0.551	0.575
AttDegree	8.064	0.742	0.609	0.427	0.460
AttGlobalContext	3.125	0.904	0.781	0.874	0.864
AttLearnableGC	2.055	0.916	0.804	0.903	0.887
SimGNN	**1.509**	**0.939**	**0.830**	**0.942**	**0.933**

Gated Graph Attention Neural Network (GGANN)



GGANN

- Input
  - Abstract Syntax Tree (AST), Function Call Graph (FCG), Data Flow Graph (DFG)
  - Integrated into FDA Graph
- Message Passing and Propagation with attention mechanism
- Aggregation to graph embedding
- Future Directions
  - Expansion to more languages and tasks
  - Accelerating training using pooling and sampling

Graph Matching Networks (GMN)

- Use GMN to evaluate GED jointly rather than computing independent graph embeddings
- Node updates consider edge and node matching between pairs
- Learning constrained as to encourage expected behavior of representation space
  - Dissimilar graphs are farther away, similar graphs nearer
- Resultant graph embedding is utilized for similarity computation.
- Future Directions:
  - Operates pairwise (no outright querying)
  - Reduce cost of computation

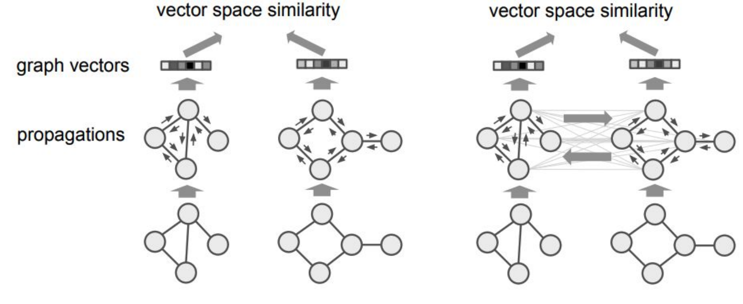
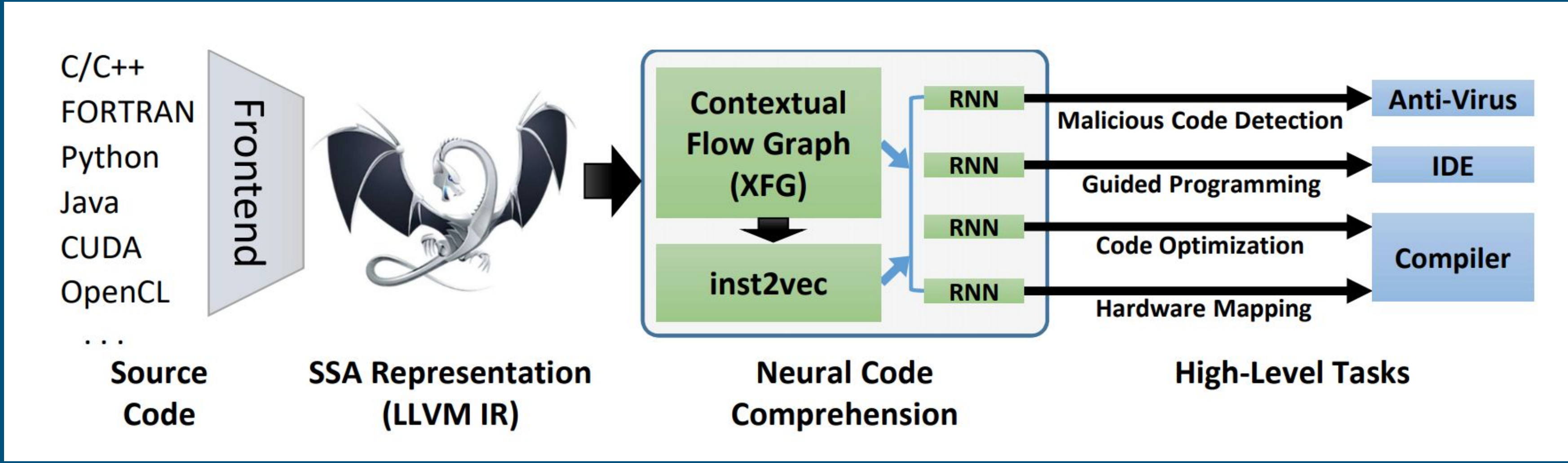


Figure 2. Illustration of the graph embedding (left) and matching models (right).

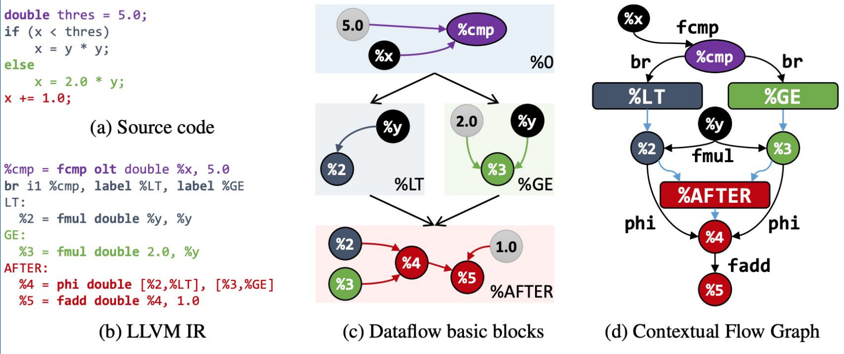
inst2vec

- Learnable representation of code semantics
- Useful for many downstream tasks



inst2vec

- conteXtual Flow Graphs (XFGs) are directed multigraphs that provide a notion of context, where nodes (variables or label identifiers) can be connected by more than one edge (data-dependence or execution dependence).



inst2vec

- Preprocess LLVM IR statements
- Neighboring statement pairs generated on which inst2vec is trained
  - Skip-gram model
- Future Directions:
  - Potential refinement via part-based models
  - Modified Differential Neural Computer rather than DNN

```
store float %250, float* %82, align 4, !tbaa !1
%10 = fadd fast float %9, 1.3
%8 = load %"struct.aaa"*, %"struct.aaa"** %2
```

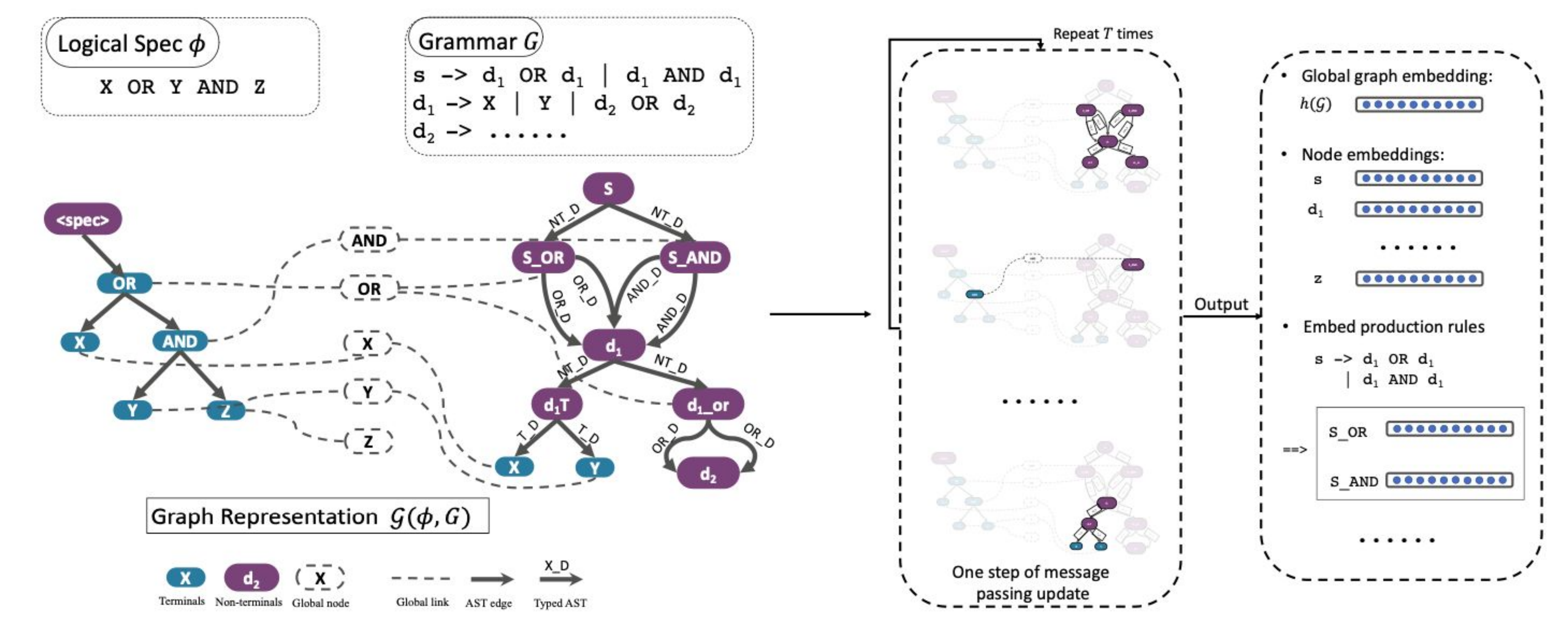
(a) LLVM IR

```
store float %ID, float* %ID, align 4
%ID = fadd fast float %ID, <FLOAT>
%ID = load { float, float }*, { float, float }** %ID
```

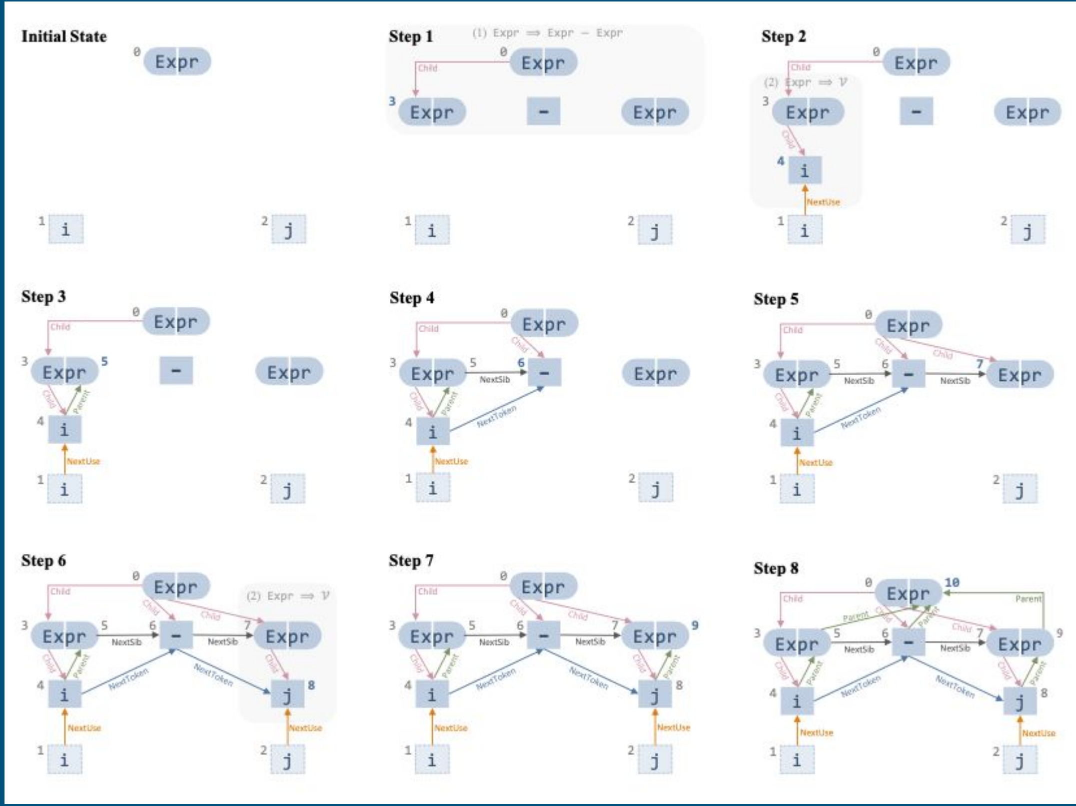
(b) inst2vec statements

Program Synthesis

Syntax-Guided Synthesis (SyGuS)



Generative Code Modeling With Graphs (ExprGEN)



Bug Detection

Decompilation

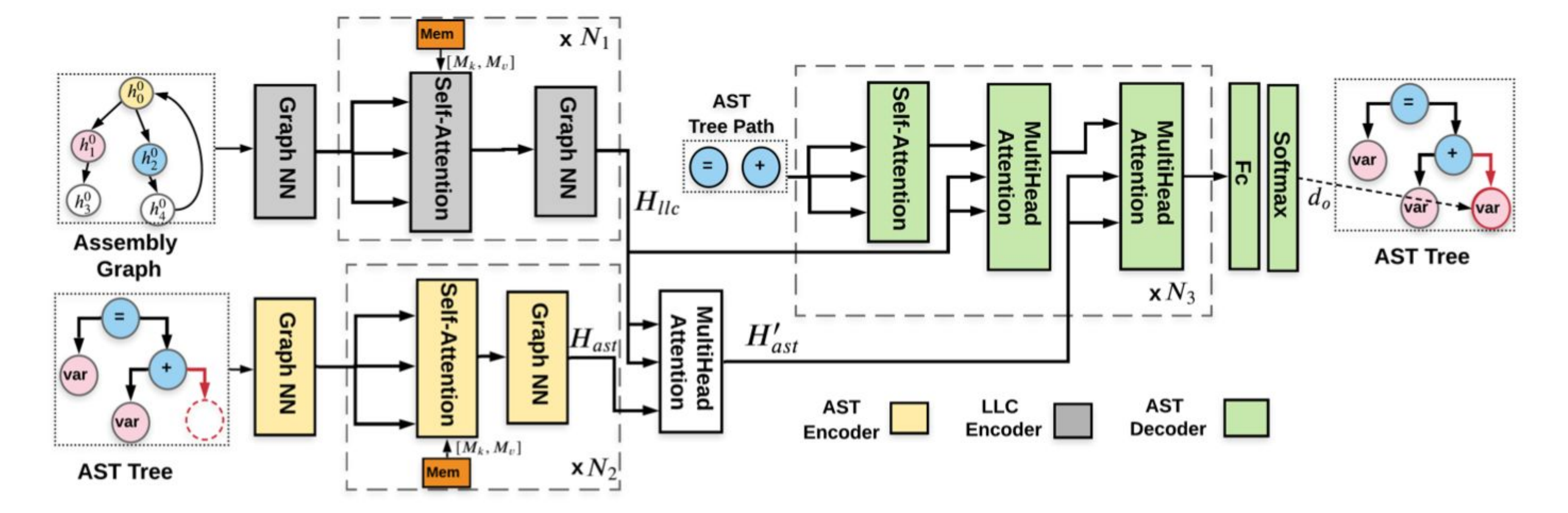
Introduction

- Decompilation is the process of obtaining the high level source code from the compiled low-level machine instruction code
- Traditional decompilers have existed for a long time, but they are mostly rule based and use pattern matching.
- At a high level, decompilation can be thought of as Machine Translation
- This comparison gave rise to research in Neural Decompilation and we'll cover the major methods in the field.

Chronology of Neural Decompilation Methods

Name	Citation	Year	Phases	Model Architecture	Model Layer	GNNs?
Katz et al. [40] RNN	[40]	2018	1	Encoder-Decoder	RNN	No
TraFix	[41]	2019	2	Encoder-Decoder	RNN	No
Coda	[26]	2019	2	Encoder-Decoder	Attention + LSTM	No
NBref	[4]	2021	2	Transformer	Self-Attention + GNN	Yes

GNN based Approach

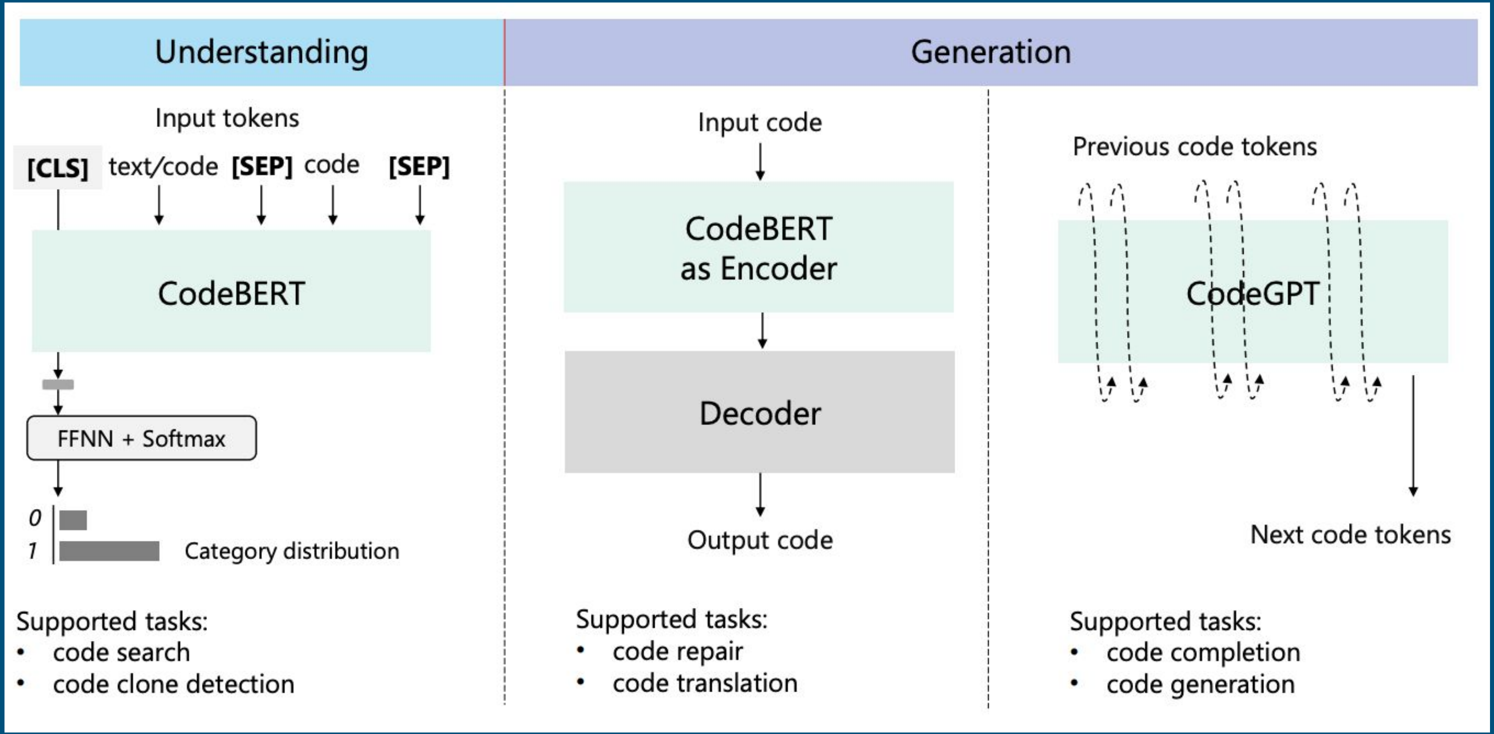


CodeXGlue

Introduction

- A benchmark dataset and open challenge with leaderboard involving cross domain tasks across Programming Languages and NLP
- 14 datasets and 10 diversified tasks
    - code-code
    - text-code
    - code-text
    - text-text
- Public Leaderboard with 3 baselines

Baselines


Understanding
Generation
Input tokens
[CLS] text/code [SEP] code [SEP]
CodeBERT
FFNN + Softmax
0
1
Category distribution
Supported tasks:
• code search
• code clone detection
Input code
CodeBERT as Encoder
Decoder
Output code
Supported tasks:
• code repair
• code translation
Previous code tokens
CodeGPT
Next code tokens
Supported tasks:
• code completion
• code generation

GNNs and future directions

- Baselines are derived from popular transformer models and treat Code as text
- Adding GNNs in the pipelines can massively boost performance on the leaderboard
- Ideas:
    - GNN Node Representations as input instead of token embeddings
    - Graph Representations for entire programs for similarity based tasks
    - Combination of Code embeddings using GNNs and Text embeddings

Thank you!

Questions?