Analyzing Software using Deep Learning

Summarizing Programs with Convolutional Networks (Part 2)

Prof. Dr. Michael Pradel

Software Lab, University of Stuttgart

Summer 2020

Overview

- **Convolutional networks**
  - Motivation and basics
  - Properties
  - Pooling
- **Tree convolution for program classification**

  Based on ”Convolutional Neural Networks over Tree Structures for Programming Language Processing” by Mou et al., 2016

Convolution of Programs

- Convolution exploits hierarchical structure of input data
- Programs are hierarchical data
  - Coarse-grained level:
    Projects – Packages – Classes – Methods
  - Code level:
    Abstract syntax trees
- Idea: Use convolution to identify important features in code

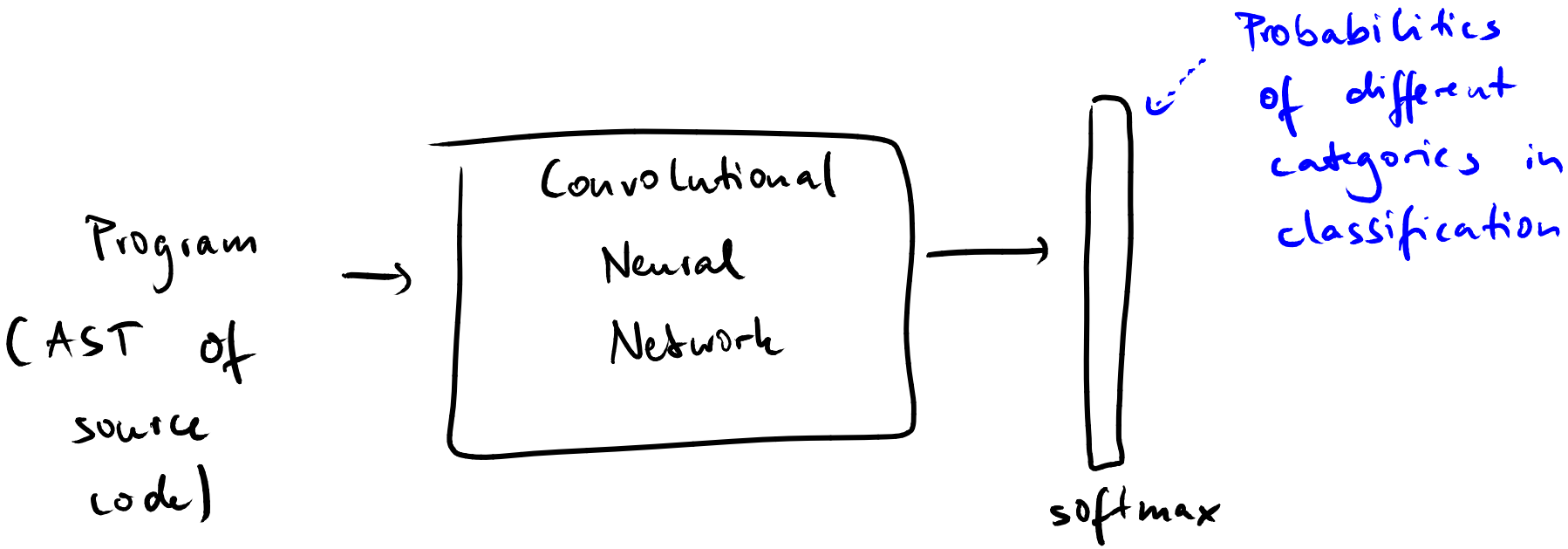
Applications

Various possible applications

Today: Classification

- Various possible ways to classify programs
  - Project where code comes from
  - Author who wrote the code
  - Instances of bug patterns
  - Malware vs. benign code
- Here: Identify functionality of code

Overview

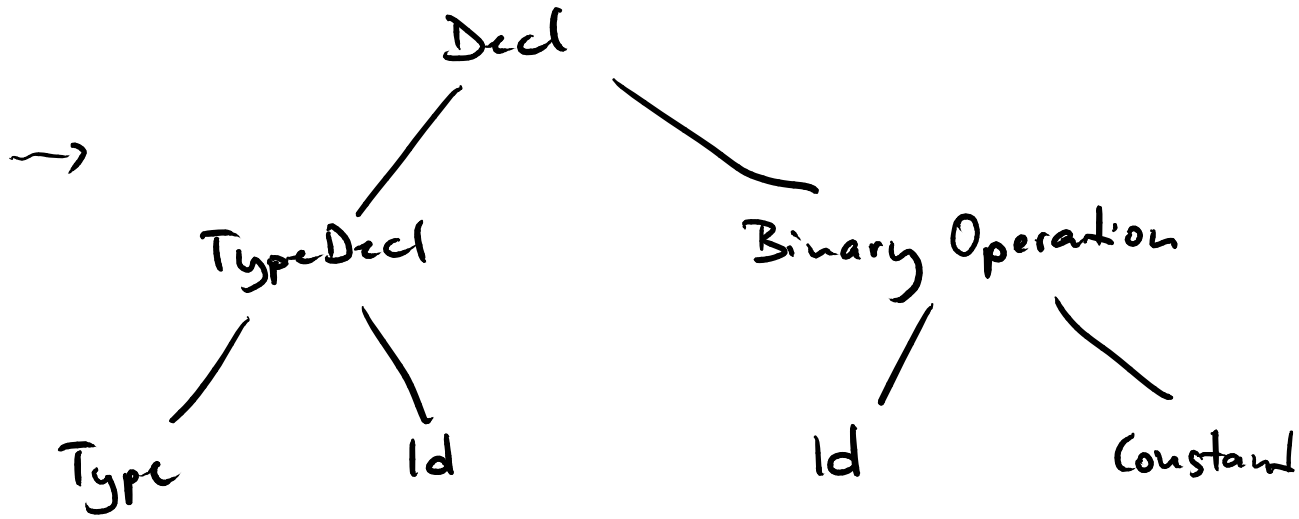

Program
(AST of
source
code)
Convolutional
Neural
Network
softmax
Probabilities
of different
categories in
classification

Tree Representation

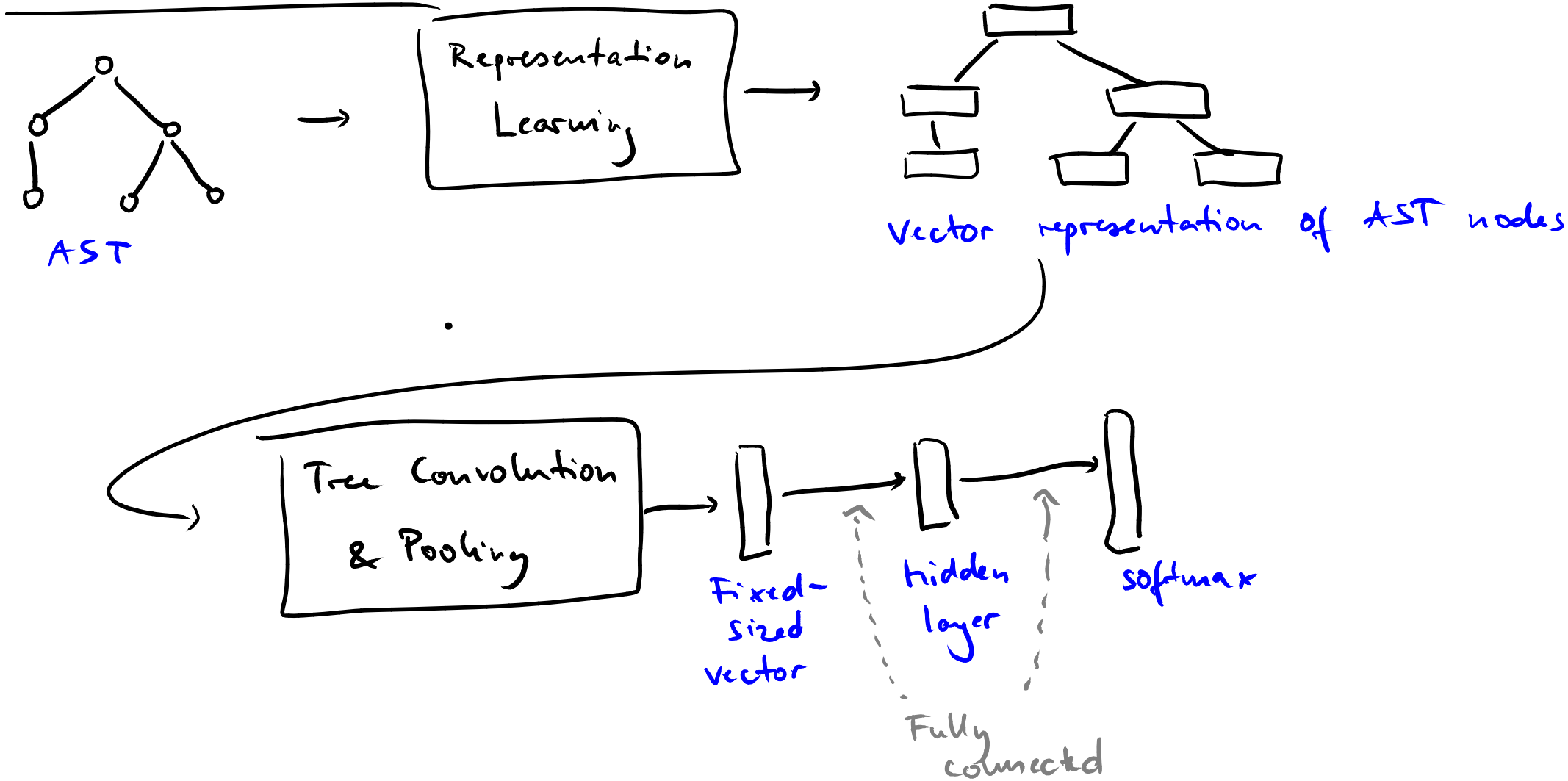
- AST where each node has at most <u>two</u> children
  ("continuous binary tree")

Example:

int a = b + 3;

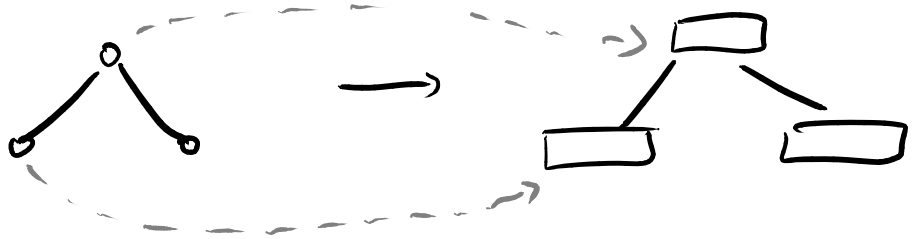


Detailed Overview



Representation Learning

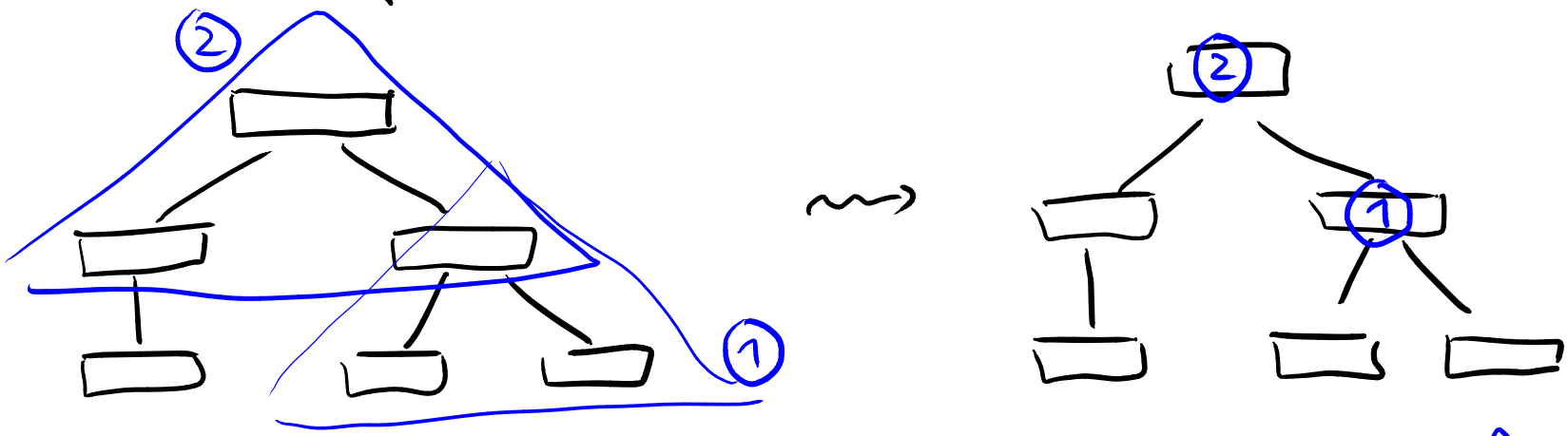
- Represent AST node as fixed-sized vector ("embedding")



- Similar nodes should have similar vectors
  - ↳ E.g., "while" ≈ "for" but "while" ≉ "constant"
- Learned in separate pre-training step

Tree-Based Convolution

- Input: Tree of vectors where nodes represent AST nodes
- Output: Tree of vectors where nodes summarize features of their children



- Idea: "Move" fixed-depth feature detector over tree

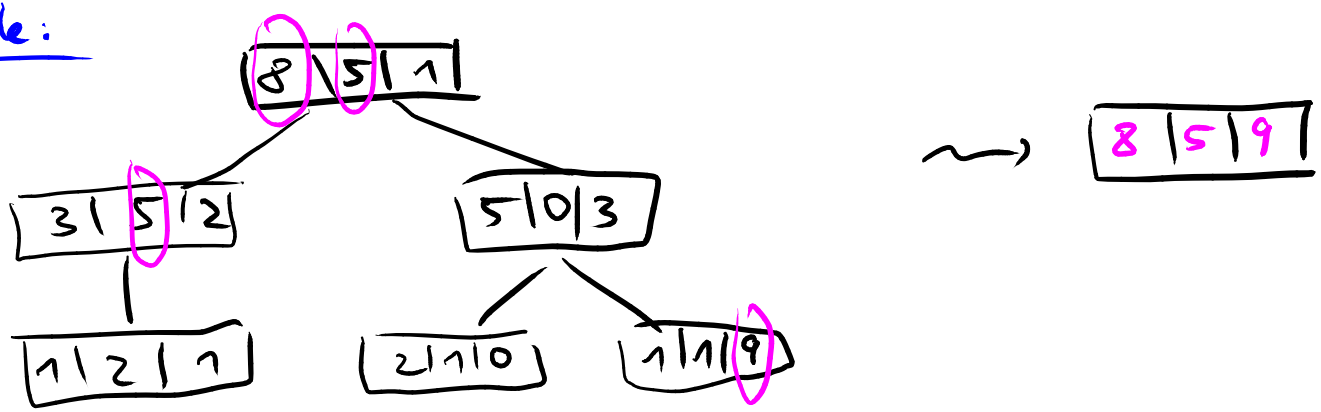
- Convolution: $y = \tanh\left(\underbrace{W_{conv}} \cdot x_{top} + \underbrace{W_{left}} \cdot x_{left} + \underbrace{W_{right}} \cdot x_{right} + \underbrace{b_{conv}}\right)$

Weights + biases of kernel

Pooling

- Before pooling: Tree of fixed-size vector but with varying nb. of nodes
- Want: Single fixed-size vector
- Here: Fixed-size pooling
  ↳ Use maximum value for each dimension

Example:



Applications

Identify functionality of a given program

Scenario 1: One out of N

- Which of 104 programming tasks has been solved?

Scenario 2: Binary classification

- Does the code contain bubble sort?

Scenario 1: One out of N

- Data: Solutions submitted to online programming education platform
- 104 problems, 500 solutions for each
- Split by 3:1:1 for training, validation, and testing
- Overall result: 94% accuracy

Scenario 2: Binary classification

- Assumption: Bubble sort is inefficient and should be avoided
- Goal: Find instances of bubble sort in given code
- Training
  - 109 programs that implement bubble sort
  - 109 programs that implement something else
- Evaluation
  - Inject bubble sort code snippet into 4,000 other programs
  - 8,000 programs in total
- Overall result: 89% accuracy

Summary

Convolutional neural networks

- Train kernel to exploit hierarchical structure of input data
- Sparse interactions
- Parameter sharing
- Equivariant representations

Application

- Tree-based convolution
- Classification of programs