Analyzing Software using Deep Learning

Lecture 5:
Introduction to Course Project

Prof. Dr. Michael Pradel
Software Lab, TU Darmstadt
Plan for Today (Part 1)

- 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
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: \[ y = \tanh (W_{conv} \cdot x_{top} + W_{left} \cdot x_{left} + W_{right} \cdot x_{right} + b_{conv}) \]
Pooling

- Before pooling: Tree of fixed-sized vectors but with a varying number of nodes
- Want: Single fixed-sized vector
- Here: Fixed-size pooling
  1) Use maximum value of 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