

Analyzing Software using Deep Learning

Lecture 3:

RNN-based Code Completion and Repair (Part 2)

Prof. Dr. Michael Pradel

Software Lab, TU Darmstadt

Plan for Today (Part 1)

- **Deep learning basics**

- Finish up last lecture

- **Recurrent neural networks (RNNs)**

- **Code completion with statistical language models**

Based on PLDI 2014 paper by Raychev et al.

- **Repair of syntax errors** ←

Based on "Automated correction for syntax errors in programming assignments using recurrent neural networks" by Bhatia & Singh, 2016

Motivation

- **Given: Program with syntax error**
- **Goal: Find a fix that removes syntax error**
- **Possible application context:
MOOCs with automated feedback on programming tasks**

Example (1)

```
def recPower (base , exp) :  
    if exp <= 0:  
        return 1  
    return base * recPower (base , exp - 1
```

Example (1)

```
def recPower (base , exp) :  
    if exp <= 0:  
        return 1  
    return base * recPower (base , exp - 1)
```



Example (2)

```
def recurPower (base , exp) :  
  if exp == 0:  
    return = exp + 1  
  else:  
    return (base * recurPower (base , exp - 1))
```

Example (2)

```
def recurPower (base , exp) :  
  if exp == 0:  
    return base ←  
  else:  
    return (base * recurPower (base , exp - 1))
```

Example (2)

```
def recurPower (base , exp) :  
  if exp == 0:  
    return base ←  
  else:  
    return (base * recurPower (base , exp - 1))
```

Beware: Fix of syntax error may not be the semantically correct fix

SynFix: Overview

Syntactically correct
student submission



Leant RNN-based
model



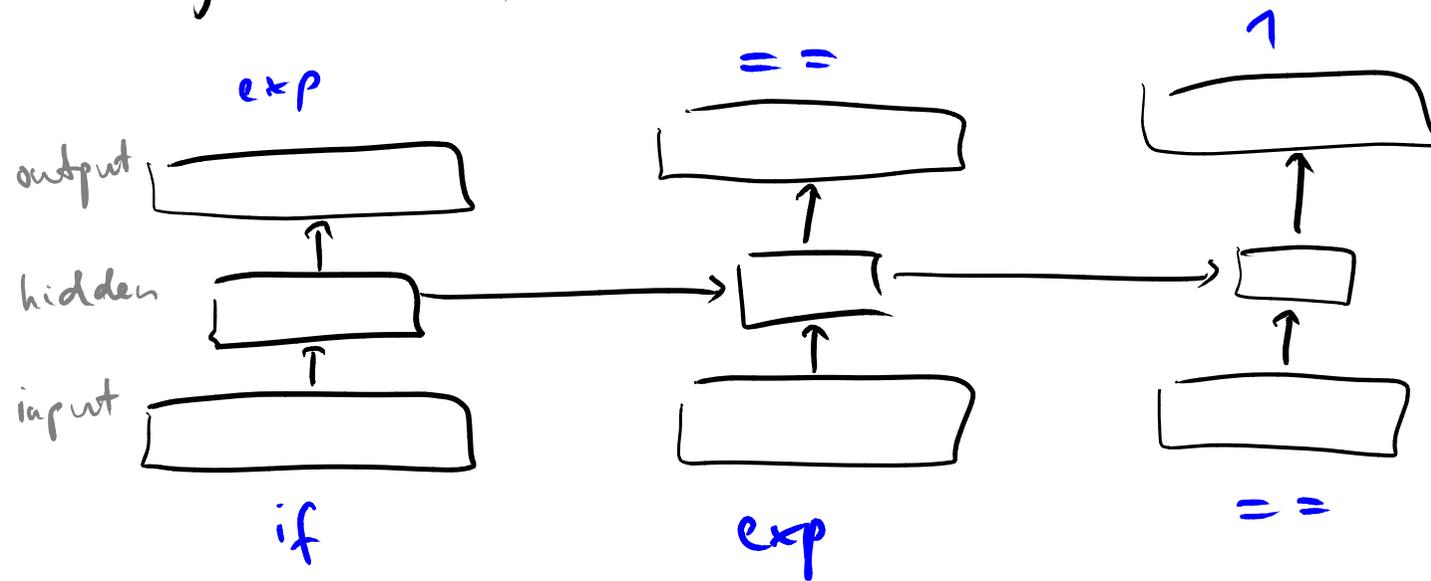
Student submission
with syntax
errors



Feedback
(= suggested fix)

RNN-based Model

- Program = Sequence of tokens



- Training:
Expected output sequence: input sequence shifted by one
- Prediction:
Provide partial program until error location & generate next token(s)

SynFix Algorithm

Given: Program with syntax error + error location

Steps:

- **Parse** and **tokenize** program
- Query network with **prefix of tokens until error location**
- Try if **inserting or replacing** one or more tokens fixes the error
- If not: Delete line with error and query network with **prefix until the error line**
- Try if **inserting** predicted tokens fixes the error

Summary

- **Recurrent Neural Networks (RNNs)**
 - Powerful class of neural networks
 - Most effective for inputs (and outputs) that are **sequences**
- Two applications
 - **Code completion:**
Predict next calls based on previous calls
 - **Repair of syntax errors:**
Predict correct tokens based on previous tokens