E0 202: Automated Software Engineering with Machine Learning

Instructor: Aditya Kanade, Indian Institute of Science, Bangalore

1 Programming assignment 2 (Announced Feb 14)

A program repair generator that can fix simple errors. The demo/evaluation will be on **Mar 14** in the regular class hours.

2 Goal

The goal of this assignment is to design a simple machine-learning based program repair solution for a toy programming language. Through this assignment, you will learn how to:

- preprocess data for machine learning tasks,
- use a deep/machine learning framework and finally,
- design, train and evaluate an encoder-decoder based neural network.

3 Problem Statement

Consider a programming language that consists of:

- arithmetic expressions, a, e.g. $1 + 3$, $x - 0$, $1 * 3$
- boolean expressions, b, e.g. True, $1 \leq x$, True $\wedge\ 1 = 1$
- if statements, if (condition) $\{S\}$ else $\{S\}$
- while statements, while (condition) $\{S\}$, and
- statement sequencing, $S; S$

In addition, consider the following set of constraints:

- No successive nesting of if statements
- No successive nesting of while statements
- No successive nesting of boolean operators
- No successive nesting of arithmetic operators

In particular, we can characterize the language formally using the grammar below:

$$
\begin{aligned}
&a \to n \mid x \mid a + a \mid a - a \mid a * a \\
&b \to \texttt{true} \mid \texttt{false} \mid a = a \mid a \leq a \mid b \wedge b \mid \neg b \\
&S \to S; S \mid \text{if}(b)\ \{S\}\ \text{else}\ \{S\} \mid \text{while}(b)\ \{S\} \mid x = a \\
&n \ \text{ denotes a } \textit{number} \\
&x \ \text{ denotes a } \textit{variable}
\end{aligned}
$$

Here is a sample programs:

```
while (False ^ False) {
  if (False) {
    x = 7 - x ;
    x = 7 * 2
  }
  else {
    x = 7 * 9 ;
    x = x * x
  }
}
```

Now, consider the following noise model over these programs:

- Delete ;
- Delete }

Notice that these modifications will always result in an incorrect program.

4 Task

Train an encoder-decoder neural network to fix programs of this kind.

4.1 Design Decisions

You may use either a Convolutional or Recurrent Neural Network (CNN or RNN) as the input). The output mechanism of the neural network is also left to you to decide. For example, you could decide to either:

- Produce the fixed program
- Emit the position and kind of fix required

4.2 Data Augmentation

We make no guarantees that the dataset we provide is enough to train a perfect repair mechanism. You may, if you feel the need to, augment the provided data with data you generate for training.

5 Evaluation

The train and validation datasets can be downloaded from the following URL: `https://goo.gl/8YeZSp`.

Your program should take as input the input in the given format and produce an output consistent with the validation output, as provided. A secret test dataset will be used for evaluation during the demo phase.

6 Grading

10 pts You will have to explain your design decisions and code.
Marks will be received for conformance to the specification, a crash-free run on the test dataset and the ability to explain your design decisions and code.

5 pts Performance on the validation dataset (relative to best).

5 pts Performance on the secret test set (relative to best).

5 pts You will be asked to modify your network to work on a modified task.

7 Resources

1. Installing Tensorflow: `https://www.tensorflow.org/install/`
2. Tensorflow APIs that you may need to make use of:
   `https://www.tensorflow.org/api_docs/python/tf/nn/dynamic_rnn`
   `https://www.tensorflow.org/api_docs/python/tf/nn/conv1d`
3. The tf.keras module has many useful functions:
   `https://www.tensorflow.org/api_docs/python/tf/keras`
4. Computational resources in CSA:
   `http://clweb.csa.iisc.ernet.in/wiki/doku.php?id=getting_access`