Creating Local LLMs for Test Assertion Generation: A Comparative Study of Knowledge Distillation from CodeT5

G.H.Dimitrov@student.tudelft.nl

Context

Find bugs in software before code delivery

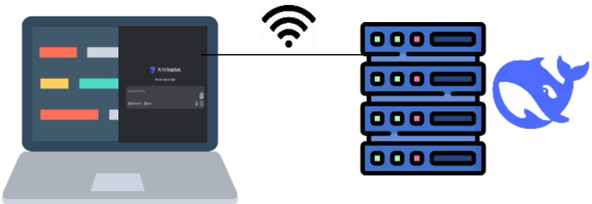
Replicate the logic that needs to be tested

Add assertions that define if the test passes of fails

Problem

```
@ParameterizedTest(name = "{index} called with: {0}")
@MethodSource(value =  "data")
void testWithStringParameter(int[] data) {
    MyClass tester = new MyClass();
    int m1 = data[0];
    int m2 = data[1];
    int expected = data[2];
    assert ???;
}
```

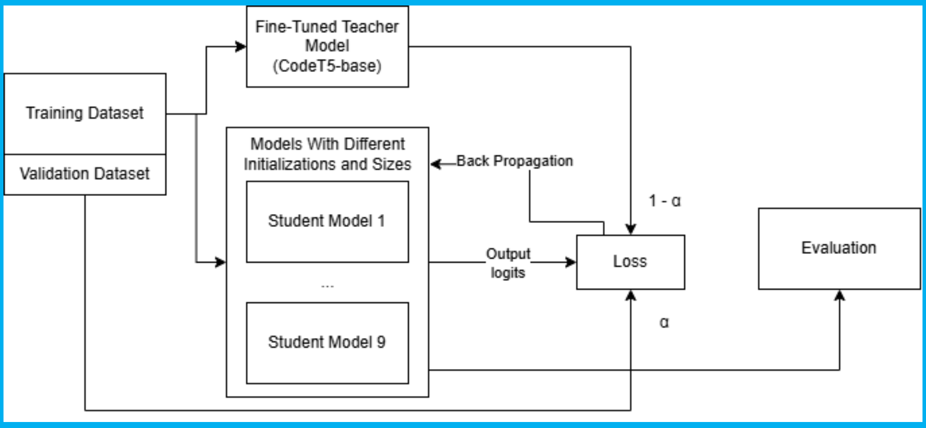
Test assertions are:
- Important for test effectiveness
- Time consuming to write manually
- Hard to generate if you do not depend on the code having the correct output to begin with



LLMs are:
- Good at understanding natural language and code
- Therefore can help generate good test assertions

But they also often are:
- Slow
- Large
- Require online connection

Bridging the Gap With Knowledge Distillation (KD)



Knowledge Distillation
- Training a student model to mimic a teacher model
- Has mostly been done for classification tasks, and not for generative ones
- Goal: Create a small model (<500MB) that is specialized for test assertion generation

Model sizes:
- 3 layers
- 6 layers
- 12 layers

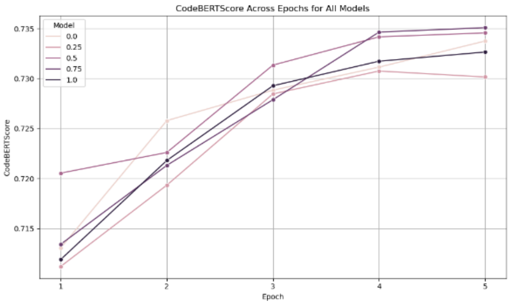
Loss function weights:
- 0.0
- 0.25
- 0.5
- 0.75
- 1.0

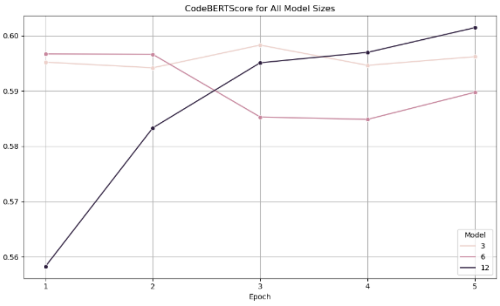
Initializations:
- Randomly initialized
- Pretrained

Specifications:
- Teacher model: CodeT5-base
- Student models: All use codet5-small architecture
- Comparison of different loss function weightings, model sizes, and model initializations on performance

Results

Model	Accuracy (%)	Similarity (%)	CodeBERTScore	CodeBLEU	Parsability rate (%)	Size (MB)	Inference speed (ms)
CodeT5-base	44.4	83.9	0.875	0.664	99.2	892	355
Student	13.4	66.9	0.734	0.395	96.2	231	118





- 83.9% of CodeBERTScore
- 25.9% of model size (231MB)
- 3x faster inference speed
- Model size seems to be the main limiting factor
- Pretrained models learn faster than randomly initialized ones
- Pretrained models perform worse than distilled models for the specific task