NomNom: Explanatory Function Names for Program Synthesizers

Amirmohammad Nazari · Souti Chattopadhyay · Swabha Swayamdipta · Mukund Raghothaman
University of Southern California

Central Problem: How to Understand Implementations Produced by Program Synthesizers?

- Program synthesizers are becoming increasingly sophisticated
- Implementations are becoming hard to understand
- **How do we help programmers understand synthesized code?**

Examples

```
f [9; 2; 7; 1] = [1; 2; 7; 9]
```

(Ellis et al., PLDI 2021)

Implementation

```
f l = map (λn. g1 l (1 + n)) (range (len l))
g1 l n = g2 (filter (λz. n > len (filter (λu. z > u) l)) l)
g2 l = hd (filter (λy. isnil (filter (λz. z > y) l)) l)
```

Generating Validated Function Names

Step ①
Recover fine-grained input-output information

```
g2 l = hd (filter (⋯) l)
```

`print l` `print out`

```
[1] ↦ 1      [2; 7; 1] ↦ 7
[2; 1] ↦ 2   [9; 2; 7; 1] ↦ 9
```

Step ②
Ask language model **LM1** to propose function name

Suggest a suitable name for a function with the following behavior: …

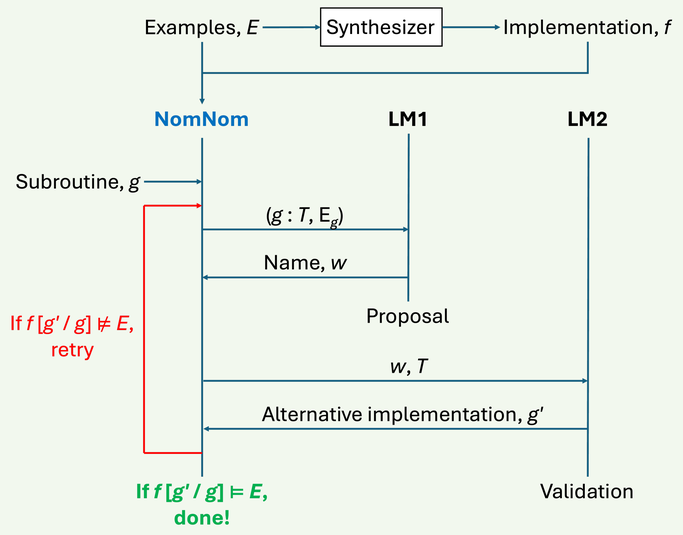
"`findLargestElement`"

Step ③
Validate proposed name using second model **LM2**

Implement a function named findLargestElement

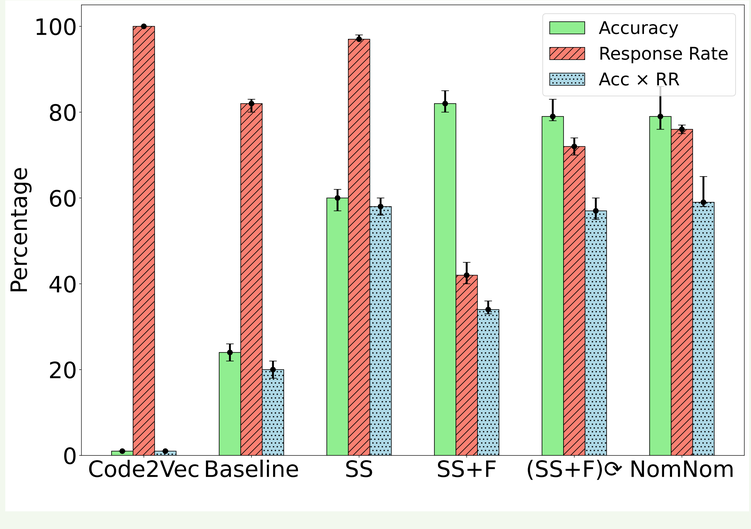
```
def findLargestElement(l):
    …
```
} `g2′`

Will `f` still work if we use `g2′` instead of `g2`?

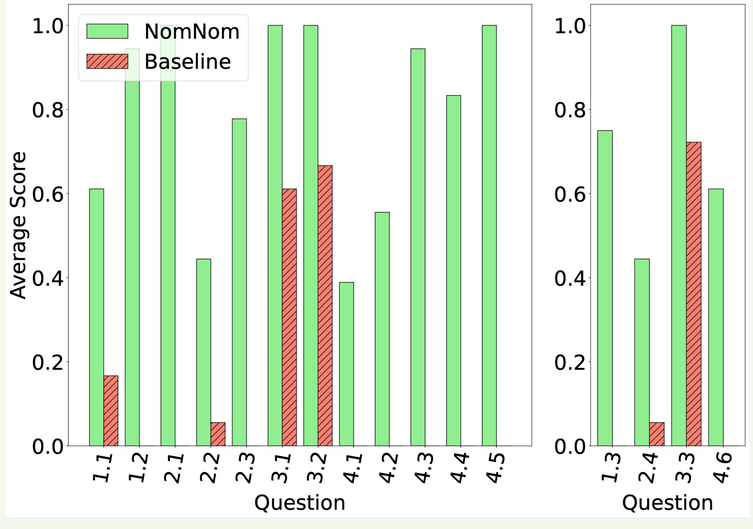


How Well Does NomNom Work?

Q1: How accurate are the generated names?



Q2: Do the generated names aid understanding?



Q3: Do users like names produced by our system?