Testing the Performance of Automated Documentation Generation with Included Inline Comments

How do comments influence the performance of the code2seq [1] model?

AUTHOR

Balys Morkūnas

SUPERVISORS

Annibale Panichella Leonhard Applis

1 MOTIVATION

- Documentation **reduces** the time needed to understand and maintain code, **lowers** the chance of code defects [2].
- But documentation takes time and resources to produce and maintain.
- Automatic Documentation Generation methods exist but still need improvement.
- Maybe they can be improved with inline comments?

2 BACKGROUND

- Code2seq - sequence to sequence model.
- It uses attention to select relevant parts of the input.
- Performs method name generation, code captioning, and code documentation.

3 HYPOTHESIS

> Inline source code comments **increase** the performance and accuracy of machine learning models for Automatic Documentation Generation

4 METHODOLOGY

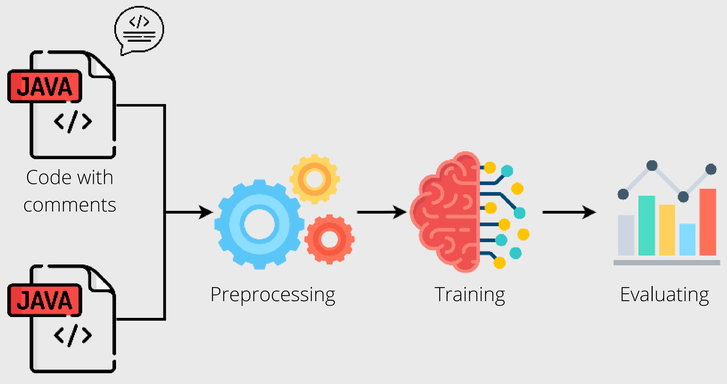

Figure 1: Pipeline for comparing performance.

5 COMMENT INCLUSION

```
1  int fooBar() {
2      // Sum two numbers
3      return 2 + 3;
4  }
```

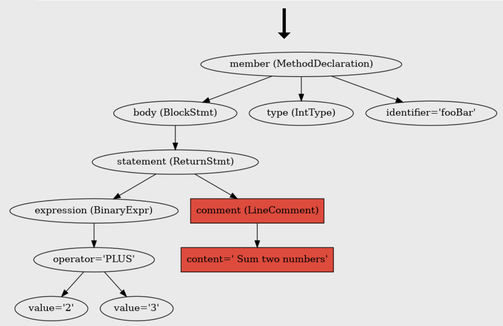

Figure 2: Comments in an AST.

- Comment filtering, stop word removal
- Comments are encoded into Abstract Syntax Trees (AST)
- AST paths are used for training the model

6 RESULTS

	BLEU	Precision	Recall	F1	Time
NOCO	6.22	0.428	0.387	0.406	11h
ICOS	5.97	0.437	0.370	0.400	**8h**
ICO	**6.28**	**0.450**	**0.391**	**0.418**	12h

Figure 3: Evaluation results for no comment, inline with stop words, and inline processed models.

Original Docstring	ICO Docstring	NOCO Docstring
buffer when possible	this method is called when the buffer is read	reads up to code len bytes from stream
gets the **children** of this **directory**	gets the list of children of the current project	returns the list of all children from the directory
returns a **host** specifier built from the provided specifier	returns the name of the host	create a host from a string
returns the **innermost** cause of code throwable	returns the root cause of the given throwable	cause the cause of the given throwable

Figure 4: Generated comment examples.

7 CONCLUSION

- Slight increase in performance for ICO model, given stop word removal (0.9% increase in BLEU score).
- More sophisticated filtering techniques.
- Comment scope detection.

REFERENCES

[1] U. Alon, S. Brody, O. Levy, and E. Yahav, "Code2seq: Generating sequences from structured representations of code," in International Conference on Learning Representations, 2019. [Online]. Available: https://openreview.net/forumid=H1gKYo09tX.

[2] N. Khamis, R. Witte, and J. Rilling, "Automatic quality assessment of source code comments: Thejavadocminer," in International Conference on Application of Natural Language to Information Systems, Springer, 2010, pp. 68–79