# A Survey on Natural Language Processing for Programming

Qingfu Zhu[1], Xianzhen Luo[1], Fang Liu[2], Cuiyun Gao[3], Wanxiang Che[1]

[1]Harbin Institute of Technology, Harbin, China

[2]Beihang University, Beijing, China

[3]Harbin Institute of Technology, Shenzhen, China

LREC-COLING 2024

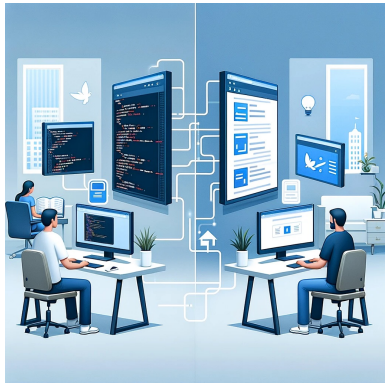
# Natural Language Processing for Programming (NLP4P)

HIT-SCIR

## What

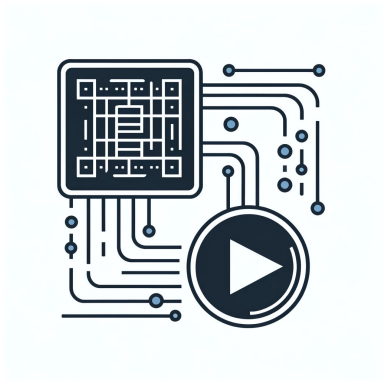

An interdisciplinary field of NLP and software engineering (SE)

## Why



Improving the productivity of both developers & non-professional users

## How



Programming language is highly structured and executable

# Structure-based & Functionality-oriented

**Structure-based**

Hierarchy Recursion Condition ⋯

**Programming Language**

```
def fib(n):
    if n == 1 or n == 2:
        return 1
    return fib(n-1) + fib(n-2)
```

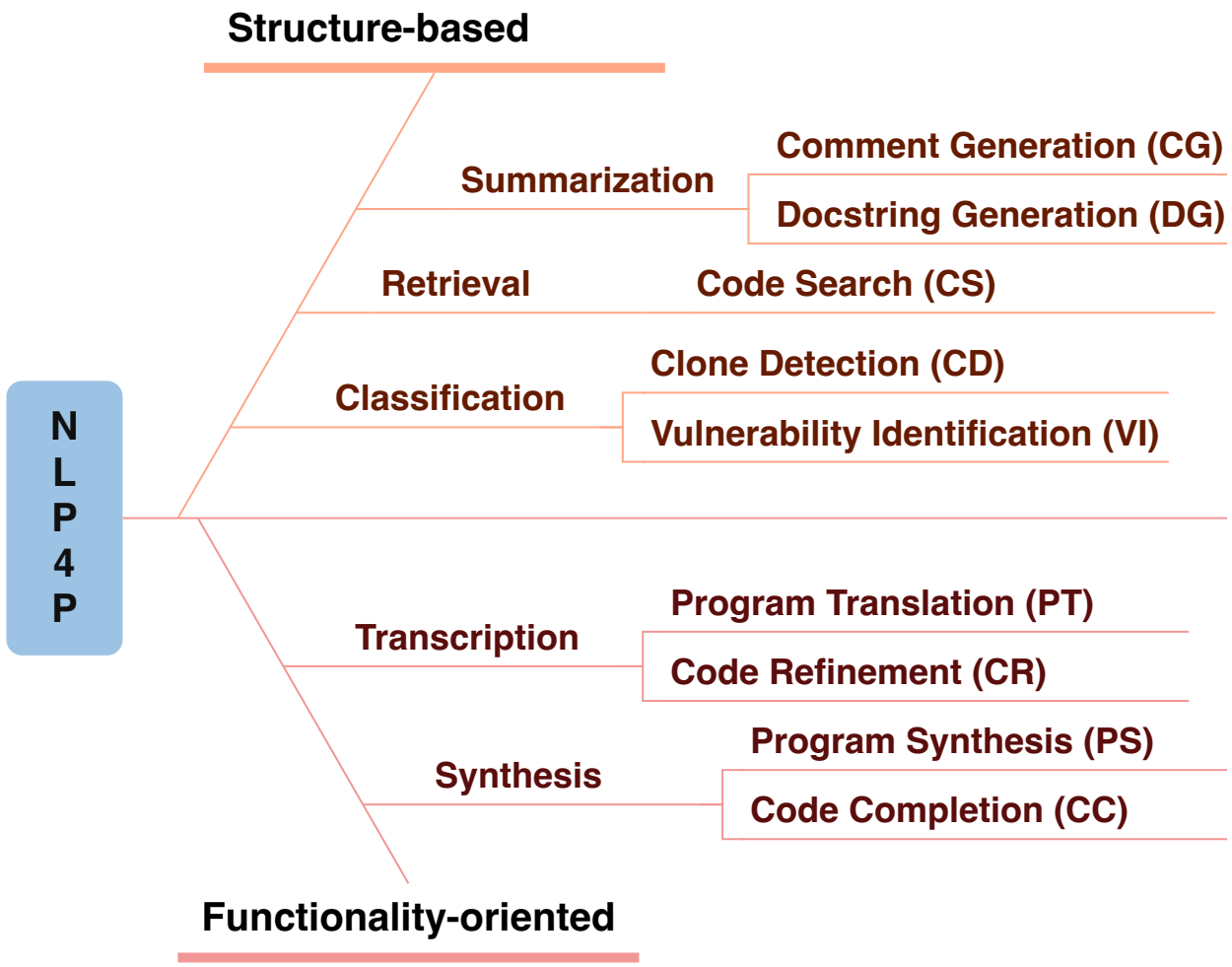
**Functionality-oriented**

| Input | Output |
|---|---|
| 1 | 1 |
| 2 | 1 |
| 3 | 2 |
| … | … |

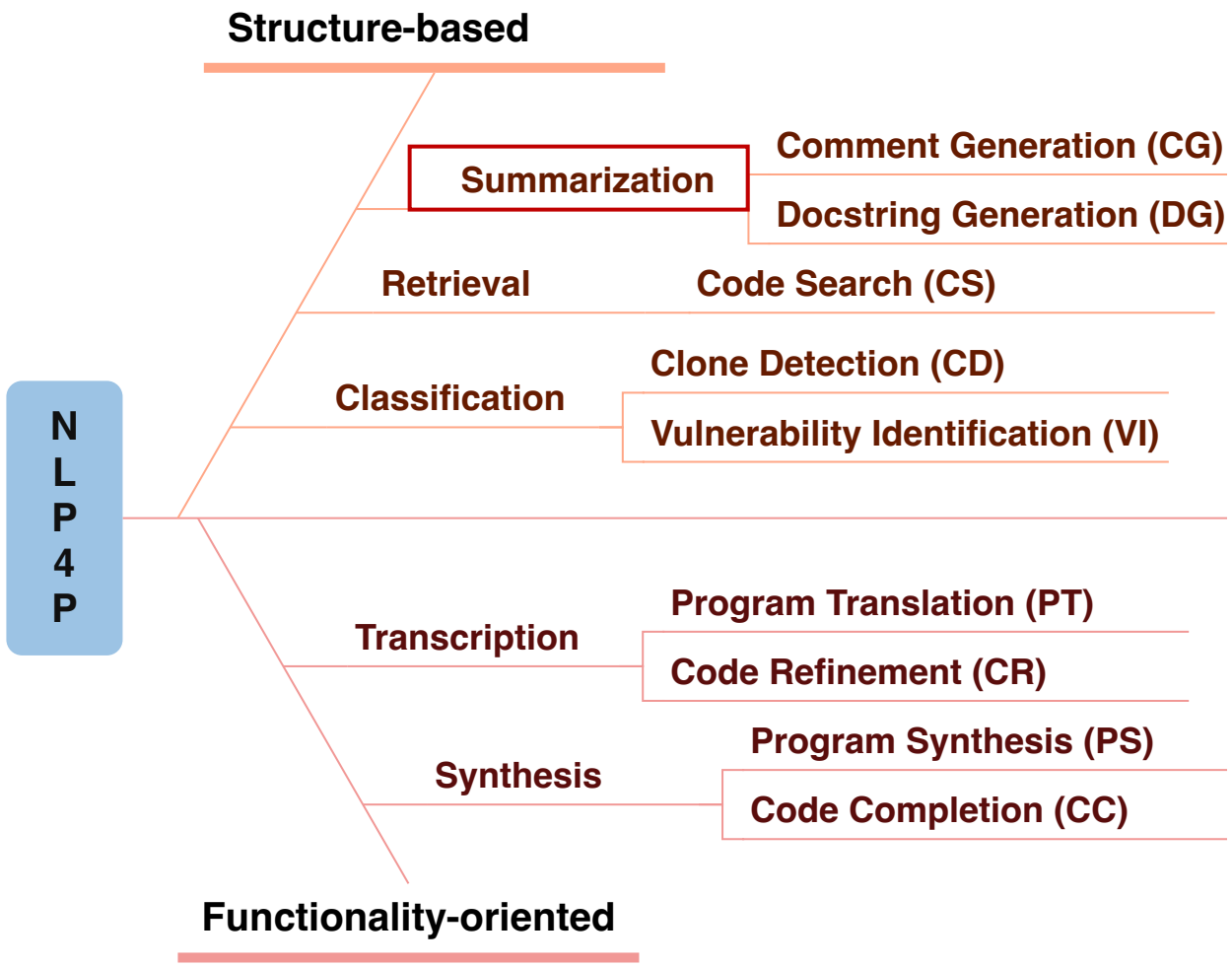
**Natural Language**

Calculate the n-th element of the Fibonacci sequence

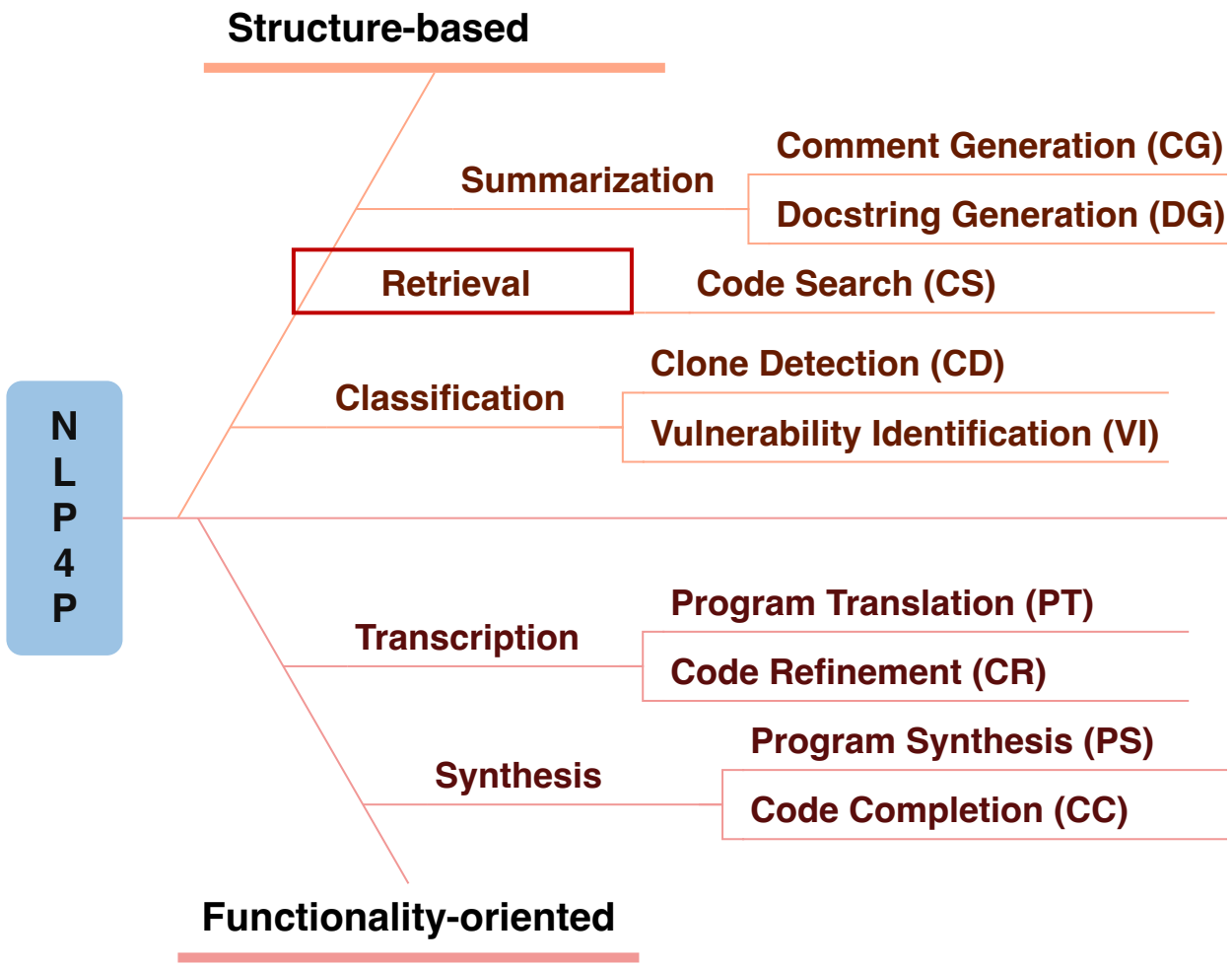
# Categories of NLP4P Tasks



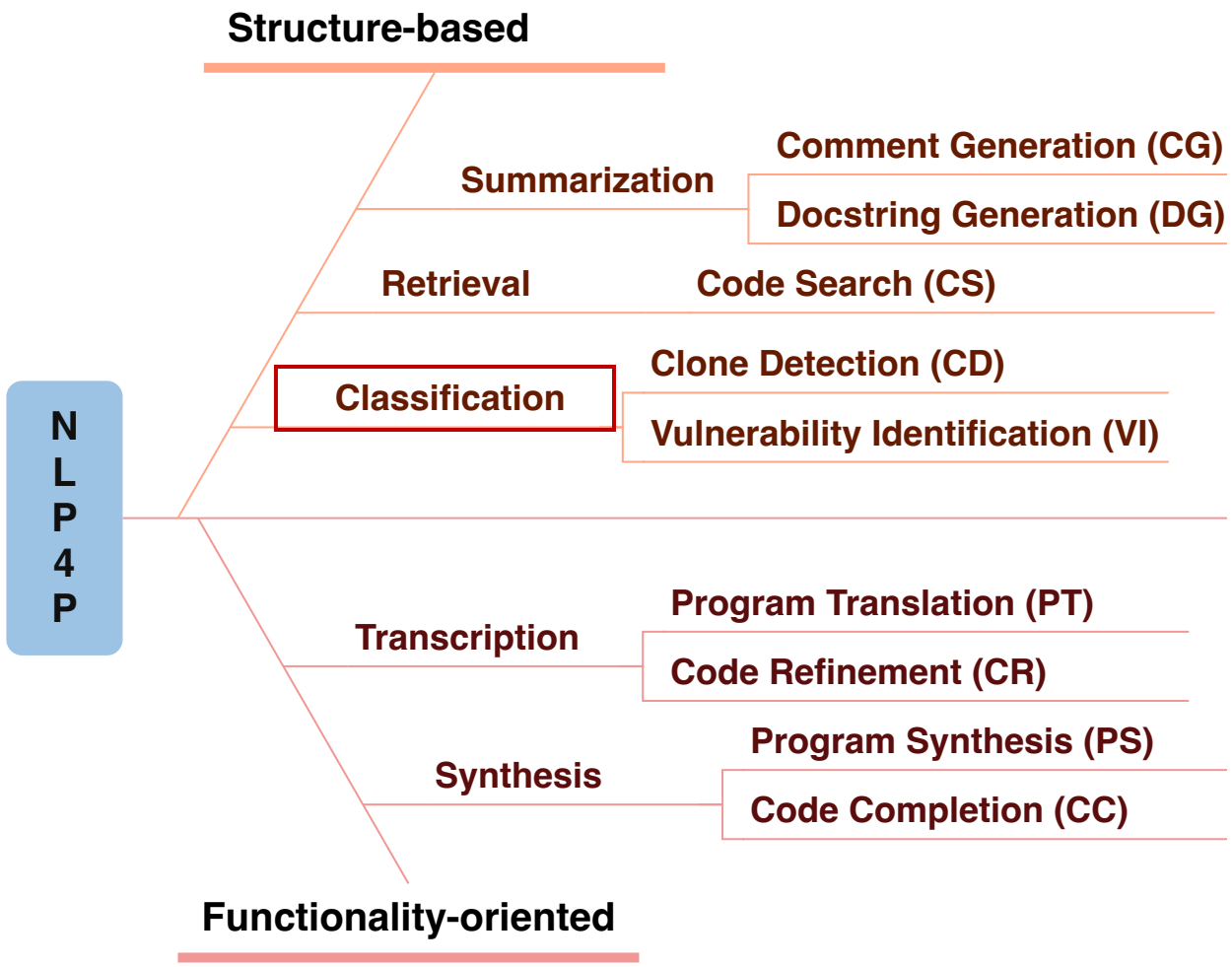
# Categories of NLP4P Tasks



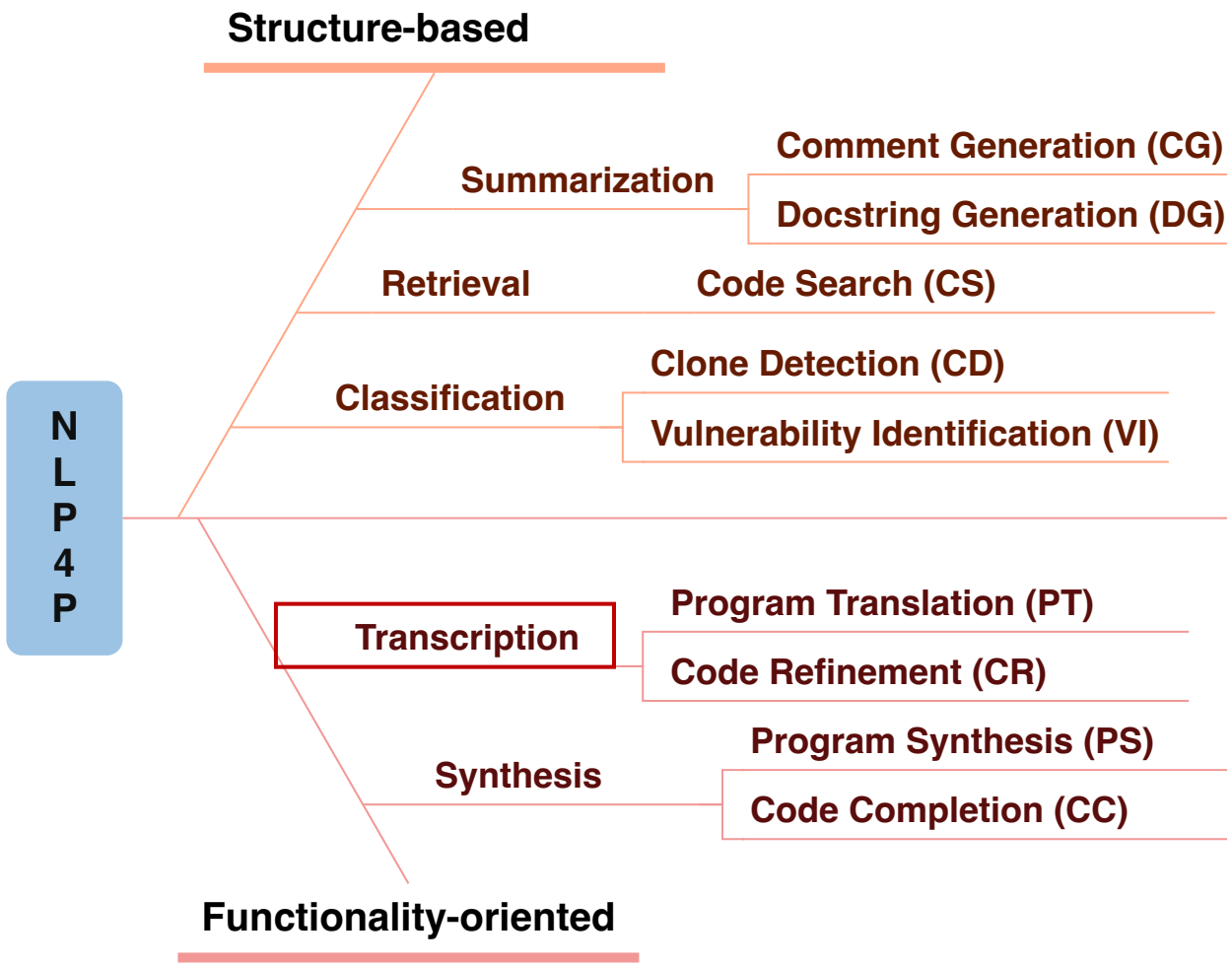
# Categories of NLP4P Tasks



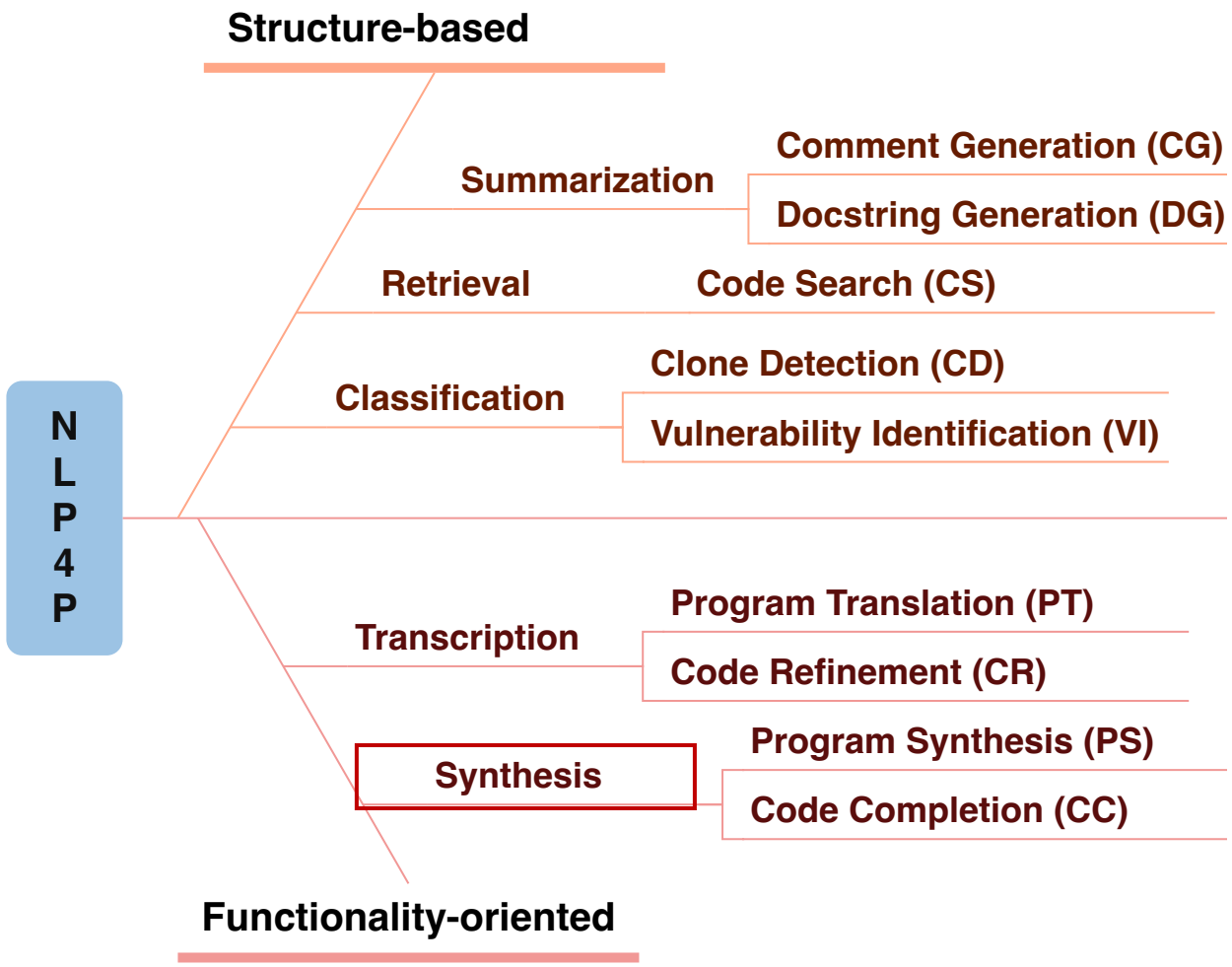
# Categories of NLP4P Tasks



# Categories of NLP4P Tasks



# Categories of NLP4P Tasks



# General Datasets

| | Data | Source | PL | Size | Type |
|---|---|---|---|---|---|
| General Dataset | PanguCoder (2022) | GitHub | Python | 147 | NL-PL<br>PL |
| | The Pile (2020) | GitHub, ArXiv,... | - | 825 | NL<br>PL |
| | Ahmad et al., 2021 | GitHub,<br>Stack Overflow | Java,<br>Python | 655 | NL<br>PL |
| | Fried et al., 2022 | GitHub, GitLab,<br>Stack Overflow | 28 | 216 | NL<br>PL |
| | The Stack | GitHub | 30 | 3,100 | PL |
| | Li et al., 2022 | GitHub | 12 | 715 | PL |
| | BigQuery | GitHub | C/C++, Go, Java,<br>JS, Python | 340 | PL |
| | BIGPYTHON (2022) | GitHub | Python | 217 | PL |
| | CodeParrot | GitHub | Python | 180 | PL |
| | Chen et al., 2021 | - | Python | 159 | PL |
| | GCPY (2022) | GitHub | Python | - | PL |

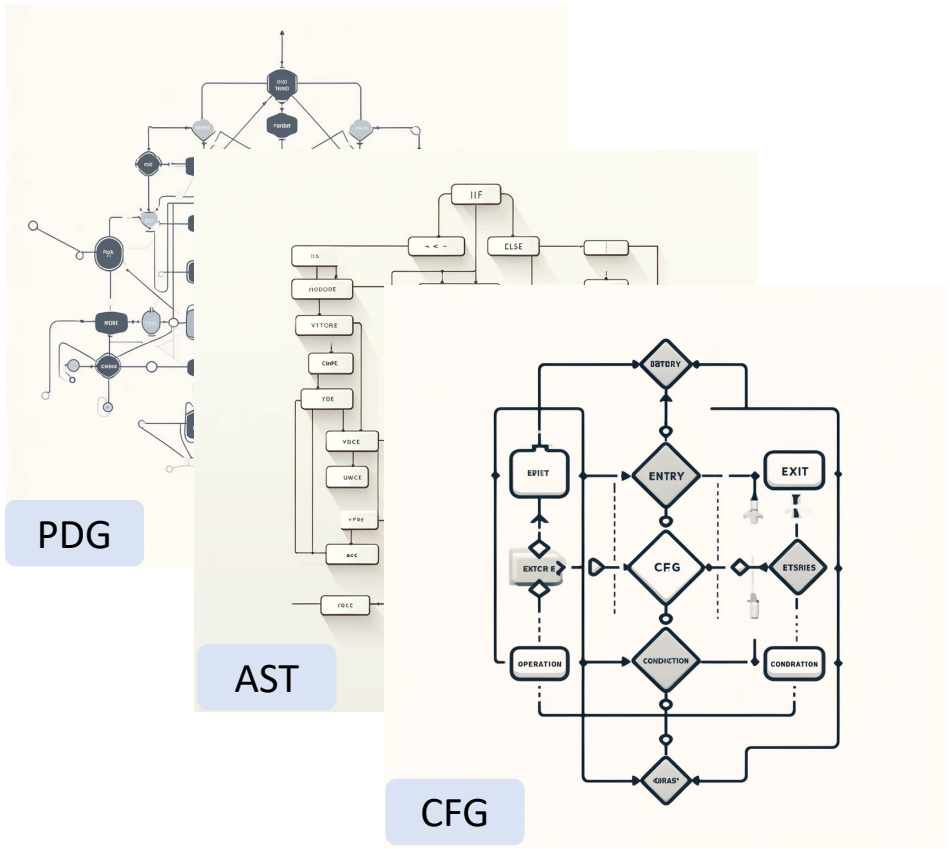
Large in scale; Noisy and non-informative

# Dedicated Datasets

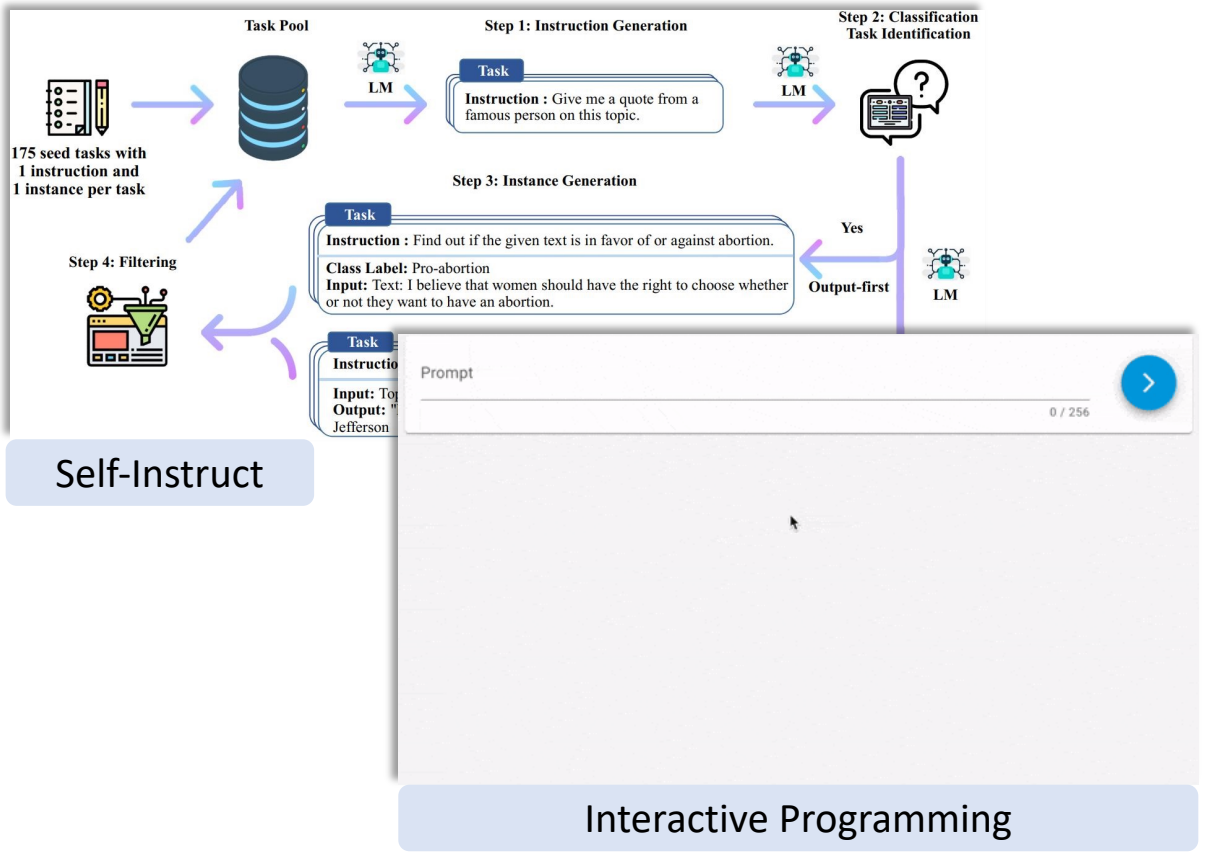
| | | Data | Source | PL | Size | Type |
|---|---|---|---|---|---|---|
| Struc-based | CG | CodeNN (2016) | Stack Overflow | C#, SQL | <1 | NL-PL |
| | CS | CodeSearchNet (2019) | GitHub | Go, Java, JS, PHP, Python, Ruby | 17 | NL-PL<br>PL |
| | CD | BigCloneBench (2014) | SeCold | Java | 2 | PL |
| | | POJ-104 (2016) | - | C/C++ | <1 | PL |
| | VI | Devign (2019) | QEMU, FFmpeg | C | <1 | PL |
| Functionality-oriented | PS | CONCODE (2018) | GitHub | Java | 13 | NL-PL |
| | | CodeNet (2021) | AIZU, AtCoder | 55 | 8 | NL-PL |
| | | CodeContests (2022) | CodeNet, Codeforces, Caballero et al., 2016 | C/C++, Java, Python | 3 | NL-PL |
| | | APPS (2021) | Codewars, AtCoder, Kattis, Codeforces | Python | 1 | NL-PL |
| | | Code Alpaca (2023) | Machine-generated | Python | <1 | NL-PL |
| | | HumanEval (2021) | Hand-craft | Python | <1 | NL-PL |
| | | HumanEval-X (2023) | Hand-craft | C++, Go, Java, JS, Python | <1 | NL-PL |
| | | MBPP (2021) | Hand-craft | Python | <1 | NL-PL |
| | | DS-1000 (2023) | Stack Overflow | Python | <1 | NL-PL |
| | | CoderEval (2023) | Github | Java, Python | <1 | NL-PL |
| | | AixBench (2022b) | - | Java | <1 | NL-PL |
| | | AixBench-L (2023a) | Github | Java | <1 | NL-PL |
| | CC | PY150 (2016) | GitHub | Python | <1 | PL |
| | | Github Java (2013) | GitHub | Java | <1 | PL |
| | PT | CodeTrans | Lucene, POI, JGit, Antlr | C#, Java | <1 | PL |
| | CR | Bugs2Fix (2019) | GitHub | Java | 15 | PL |

Small in scale; Informative, task-aware

# Techniques



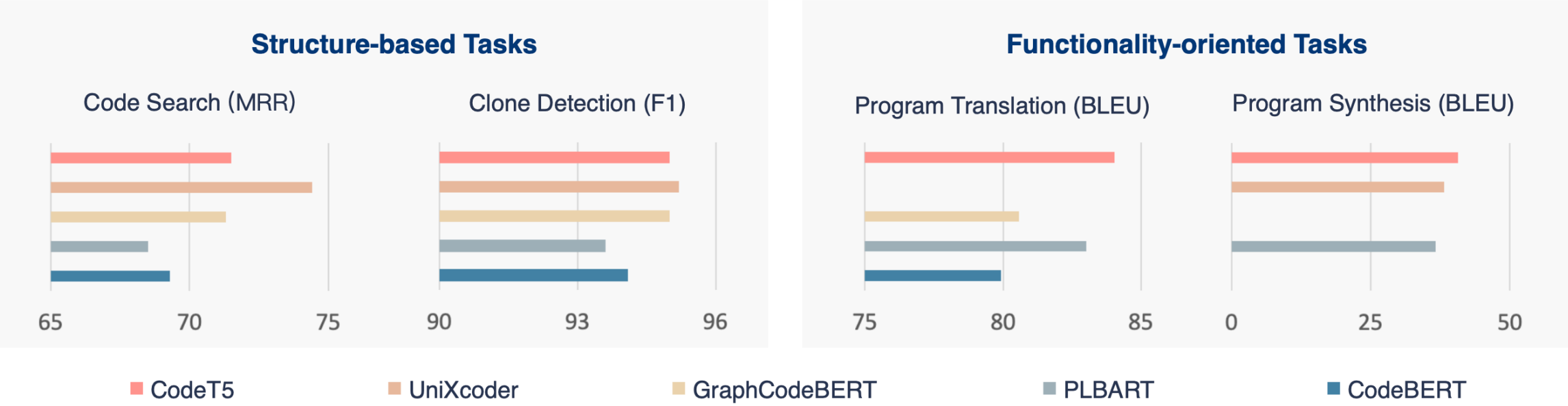

Structure-based Understanding




Functionality-oriented Generation

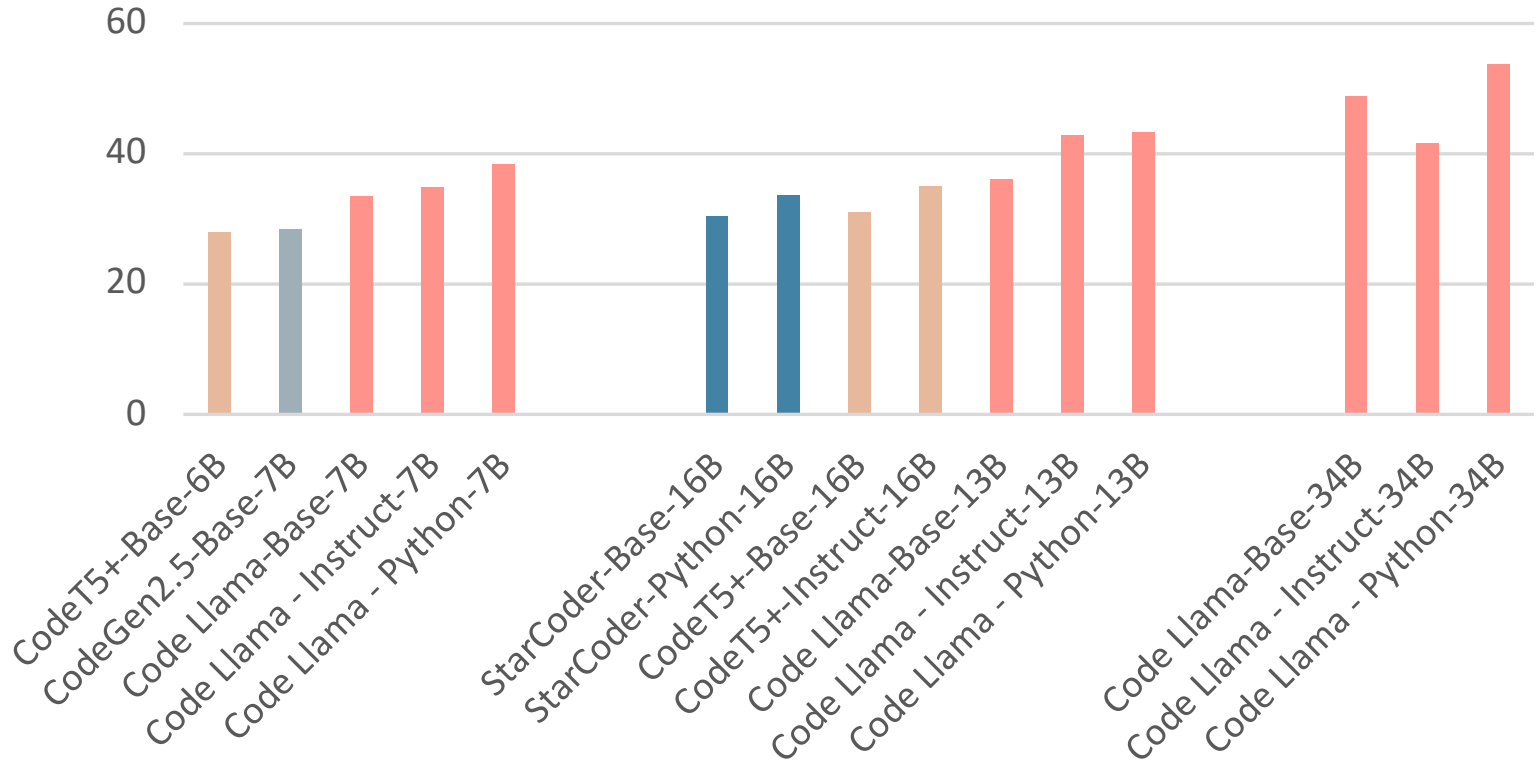
Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*.
A conversational paradigm for program synthesis. *arXiv preprint arXiv:2203.13474*.

# Representative Pre-training Models




- Incorporating structure-based representation can boost the performance
- UniXcoder performs better on understanding tasks, while CodeT5 outperforms others on generation tasks

# Representative Pre-training Models

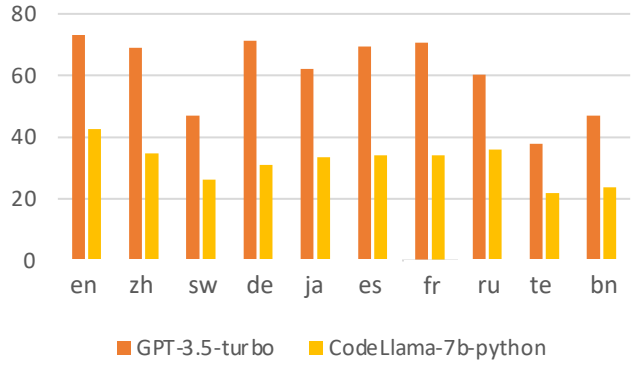



Model size is the primary factor

Constantly learning a specific PL is promising to gain improvement

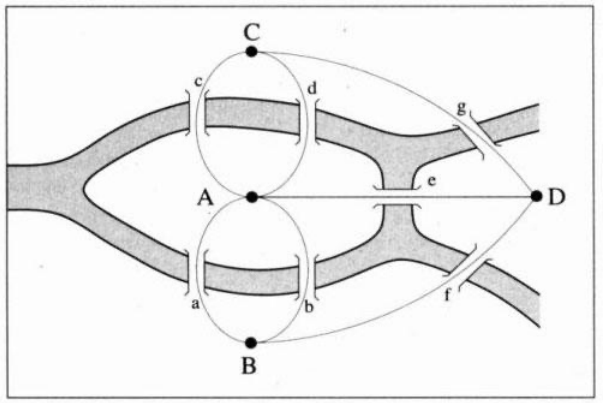
# Future Directions

## Multilingual



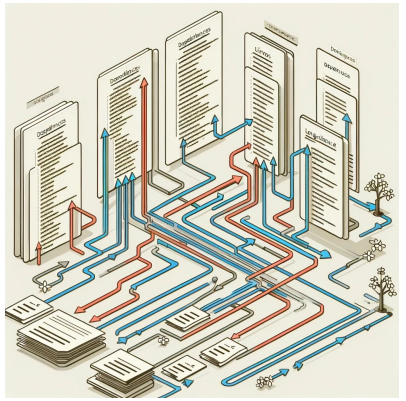

the performance of low-resource NL and PL is much worse than the average performance

## Multi-modal




NL specification may refer to other modalities for better understanding

## Long-Context



PL has a long-distance dependency across multiple lines and files

# Conclusion

- We identify two properties of PL: structure- based and functionality-oriented
- We systematically review current work, covering tasks, datasets, evaluation methods, techniques, and representative models that achieve SOTA performance.
- We illustrate unexplored areas and suggest potential directions for future work.

# Thanks and Q/A