| **Paper Title** | **Author(s)** | **Summary** | **Methodology** | **Tools** | **Future Scope** |
| --- | --- | --- | --- | --- | --- |
| Evaluating Python Static Code Analysis Tools Using FAIR Principles | Hristina Gulabovska, Zoltán Porkoláb | Evaluates Python static analysis tools using FAIR principles to enhance usability. | Analyses tool features against FAIR guidelines for effectiveness. | Various Python tools | Enhance tool usability based on FAIR principles. |
| Static Code Analyser to Enhance Developer Productivity |  | Introduces a tool to improve coding practices and developer productivity through static analysis. | Studies the impact of static analysis on code quality and error reduction. | no | Develop additional features to boost productivity further. |
| Towards More Sophisticated Static Analysis Methods of Python Programs |  | Investigates advanced static analysis methods like symbolic execution compared to traditional ones. | Compares techniques and discusses their real-world applicability. | Symbolic execution tools | Explore sophisticated methods for integration into existing tools. |
| Static Type Analysis for Python |  | Presents "Type," a tool for static type annotation and inference to enhance code quality. | Simulates built-in modules for maintaining type safety through static analysis. | Type tool | Expand type checking capabilities and integrate with other tools. |
| A Large-Scale Security-Oriented Static Analysis of Python Packages in PyPI |  | Examines security issues in Python packages using static analysis to identify vulnerabilities. | Analyses PyPI datasets to detect security flaws through static analysis methods. | PyPI dataset | Broaden security-focused analysis tools for more package coverage. |