Project Proposal: Retrieval-Augmented Generation (RAG) Based Test Generation for Java to Python Translation Validation

Track: Development Track

Team Members: Kaiyao Ke (kaiyaok2@illinois.edu)

Functions and Users: This project aims to develop a tool that automates validation of Java-to-Python code translation by dynamically constructing equivalent Python objects from Java runtime data. The tool will focus on unit test validation by capturing Java function I/O pairs and side effects, then generating corresponding Python test assertions. The tool is an extension of automated translation workflows and will benefit developers migrating large codebases from Java to Python, particularly in AI and Data Science domains where Python dominates.

Significance: Existing Java-to-Python translation tools struggle with validating translated methods in isolation because manually constructing equivalent test cases is infeasible. Our tool mitigates this by dynamically capturing runtime behavior in Java and mirroring it in Python, ensuring correctness without requiring manual test writing. This approach addresses a critical pain point in software migration, particularly for AI, ML, and analytics libraries. By improving translation validation, our tool enhances reliability and reduces the manual effort needed for code migration.

Approach:

- **Retrieval Component:** Fetch Java type documentation via web crawling and document parsing. An LLM processes retrieved data to generate Python equivalents, forming a universal type map of "standard" types (example: java.util.Collections$UnmodifiableRandomAccessList -> tuple)
- **Java Object Capture:** Utilize `AspectJ` to intercept Java object states at runtime. Extract structured JSON representations that include types, static fields, and instance attributes.
- **Python Object Construction:** Parse JSON and reconstruct equivalent Python objects using the type mapping from retrieval. Recursively set attributes to mirror Java objects accurately.
- **Test Generation in Python:** Extract function I/O pairs and side effects from Java unit tests to automatically generate Python test assertions. Implement method mocking to validate functions in isolation.

Evaluation: The tool's effectiveness will be evaluated against AlphaTrans, a state-of-the-art repository-level Java-to-Python translation tool (https://arxiv.org/html/2410.24117v1, accepted at FSE'25). This assessment will focus on improving the accuracy of isolated function validation, addressing a known limitation in the original paper.

Timeline:

- **Week 1:** Develop RAG-based Java-to-Python type mapping.
- **Week 2:** Implement Java object capture, serialization, and Python object deserialization.
- **Week 3:** Set up method mocking for isolated function validation.
- **Week 4:** Final bug fixes and tool refinement.

Task Division: I will take full responsibility for the development of the tool.