An Exploratory Study on the Roles of LLMs in Code Similarity Detection and Generation

Palash R. Roy
M.Sc. Candidate
Dept. of Computer Science
University of Saskatchewan

Supervisor
Banani Roy
Assistant Professor
Dept. of Computer Science
University of Saskatchewan

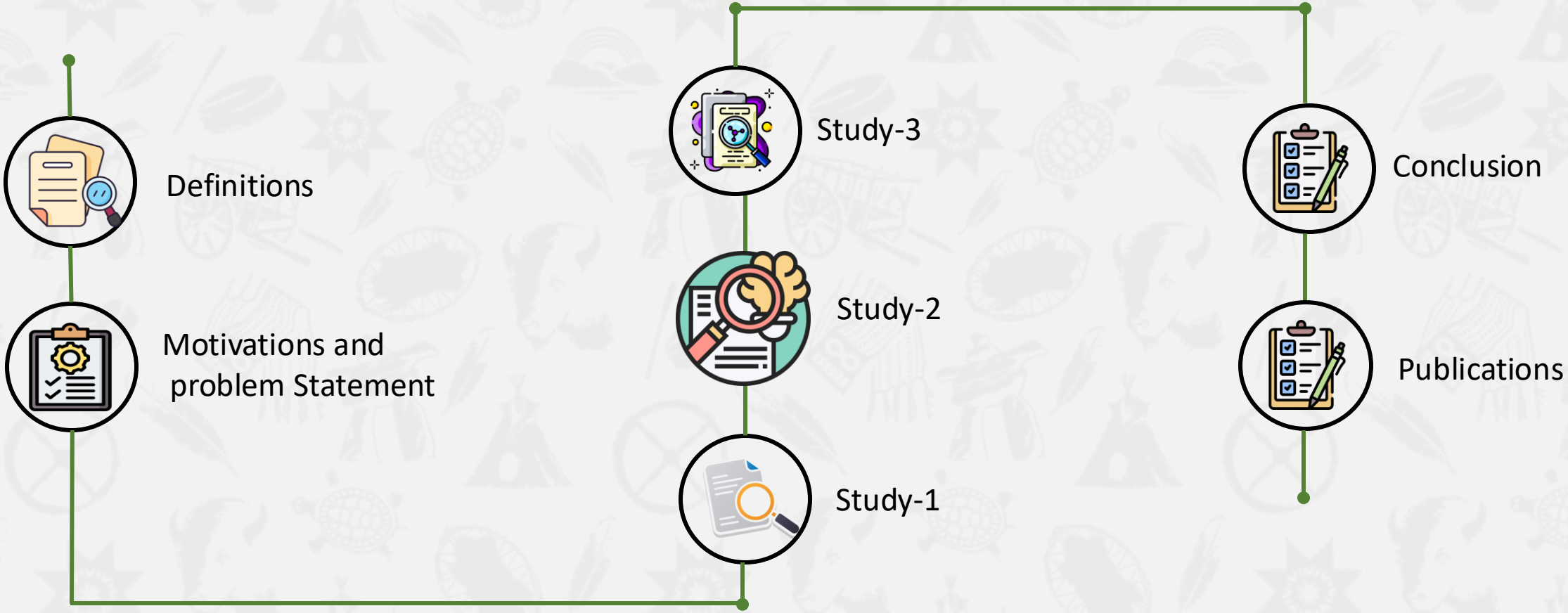
Supervisor
Kevin Schneider
Professor
Dept. of Computer Science
University of Saskatchewan

UNIVERSITY OF SASKATCHEWAN

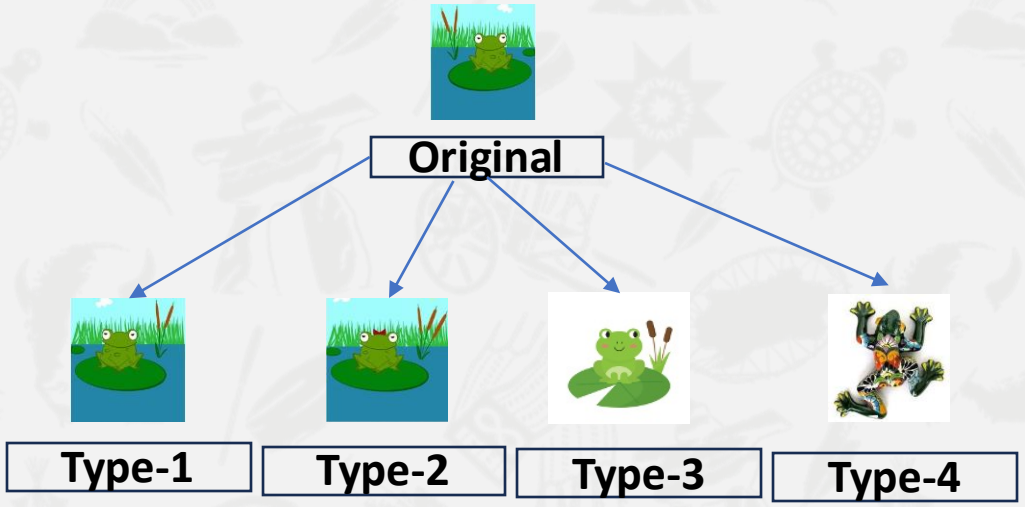
BE WHAT THE WORLD NEEDS

iSE LAB
INTERACTIVE SOFTWARE ENGINEERING LAB

Presentation Outline



Similar Codes/Clones



Our Focus

Semantic Clones

```python
def factorial(n):
    x = n
    for i in range(1,n,1):
        x = x*(n-i)
    return x
```

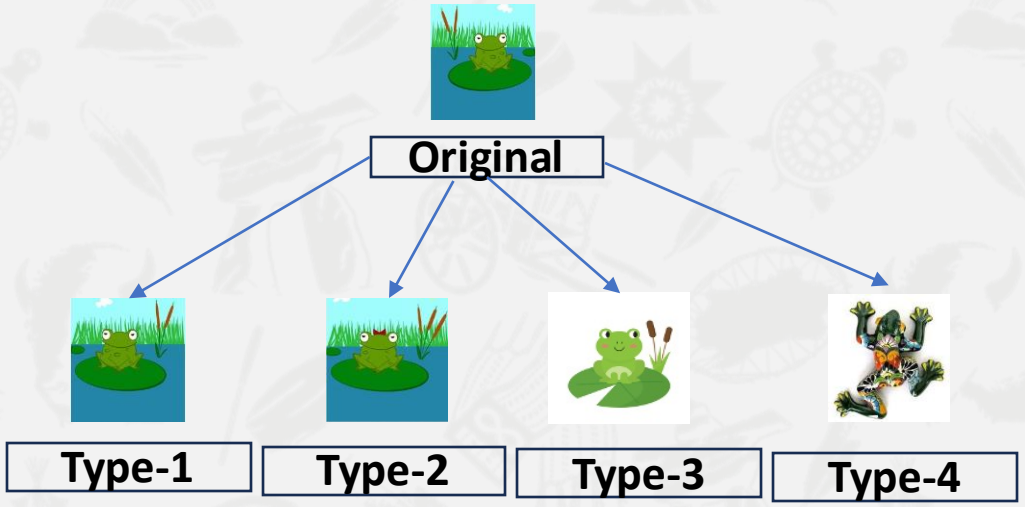
```python
def factorial(n):
    if n <= 1:
        return 1
    else:
        return n * factorial(n-1)
```

Cross-Language Clones

```java
// Java
public static boolean longerTF (boolean [] guess) {
    int variableTrue = 0;
    int variableFalse = 0;
    for (int x = 0;
    x < guess.length; x ++) {
        if (guess [x]) {
            variableTrue ++;
        } else {
            variableFalse ++;
        }
    }
    return variableTrue >= variableFalse;
}
```

```python
// Python
def longerTF(guess):
    variableTrue = 0
    variableFalse = 0
    for x in range(len(guess)):
        if (guess[x]):
            variableTrue += 1
        else:
            variableFalse += 1
    return variableTrue >= variableFalse
```

Similar Codes/Clones



Our Focus

Semantic Clones

```python
def factorial(n):
    x = n
    for i in range(1,n,1):
        x = x*(n-i)
    return x
```

```python
def factorial(n):
    if n <= 1:
        return 1
    else:
        return n * factorial(n-1)
```

Cross-Language Clones

```java
// Java
public static boolean longerTF (boolean [] guess) {
    int variableTrue = 0;
    int variableFalse = 0;
    for (int x = 0;
    x < guess.length; x ++) {
        if (guess [x]) {
            variableTrue ++;
        } else {
            variableFalse ++;
        }
    }
    return variableTrue >= variableFalse;
}
```

```python
// Python
def longerTF(guess):
    variableTrue = 0
    variableFalse = 0
    for x in range(len(guess)):
        if (guess[x]):
            variableTrue += 1
        else:
            variableFalse += 1
    return variableTrue >= variableFalse
```

Impacts of Clone

Positive Impacts *[Krinke WCRE 2007, Krinke SCAM 2008, Krinke, IWSC 2011, Hotta et al. IWPSE 2010, Gode and Harder CSMR 2011, Kapser and Godfrey EMSE 2008, Alam et al. ICSME 2023, Roy et al. IWSC 2023, Mondal et al EMSE 2017]*

- Increased productivity by code reuse.
- Easing Program Comprehension.
- Avoiding risks by reuse of mature and tested code.



Negative Impacts *[Lozano and Wermelinger, ICSM 2008 , Lozano and Wermelinger, IWSC 2010, Juergens et al. ICSE 2009, Barbour et al. ICSM 2011, Mondal et al. ACR 2012, SAC 2012]*

- Hidden bug propagation.
- Inconsistent change.
- Higher change proneness.
- Larger codebase (code bloating).
- Increased maintenance efforts.

What are the consequences?

”Toyota's Unintended Acceleration leading to serious accidents and fatalities."
2009-2010
Source: Barr Group Report, NHTSA

”The Therac-25, a radiation therapy machine, delivered massive overdoses of radiation to patients due to software errors, causing multiple injuries and deaths."
1985-1987
Source: Nancy and Clark, UofW, UofC

”NASA’s official report on the loss of the Mars Climate Orbiter, including findings related to unit mismatches and code errors.."
1999
Source: Simscale

”Trading algorithm failure and its impact on Knight Capital Group."
2012
Source: *Bloomberg*, *The New York Times*, and *Reuters*

” Ariane 5 rocket self-destructed 37 seconds after launch due to a software bug related to data conversion."
1996
Source: The European Space Agency

” Boeing 737 MAX Crashes (2018 and 2019)."
2018,2019
Source: The Newyork Times, The Wall Street Journal, ,U.S. Federal Aviation Administration

Code Clone: Friend or Foe?

To capitalize the benefits of code cloning while avoiding the negative impacts, researchers commonly agree that :

"Clones need to be managed properly"

– CK Roy et al. CSMR-WCRE 2014, Vision Keynote



Why Semantic Clone?

Among all the clone types, semantic or cross-language clones are harder to detect *[Al-Omari et al IWSC 2020, Baxter et al. ICSM 1998, Roy et al Queens 2007, Wu et al ASE 2020,* Svacina et al SAC 2020*]*.

Challenges :

- Syntactic Differences.
- Structural Variations.
- Variable names and Code Refactoring.

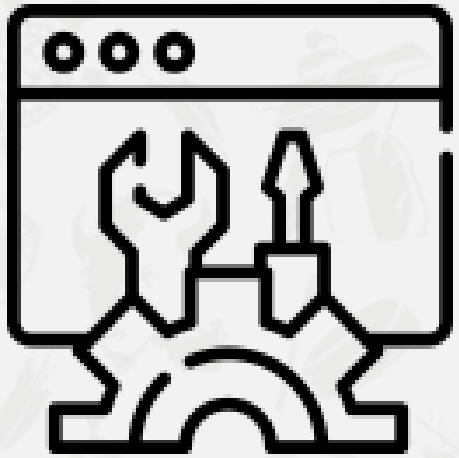


Impact:*[Al-Omari et al IWSC 2020, Baxter et al. ICSM 1998, Roy et al Queens 2007, Wu et al ASE 2020,* Svacina et al SAC 2020*]*.

- Facilitate code reuse, bug detection, maintainability across languages.
- Contribute to advanced code search and refactoring tools.

Clone Detection Techniques

- Textual Approaches.
- Lexical Approaches.
- Tree-based Approaches.
- Graph-Based Approaches.
- Metrics-based Approaches.
- ML or DL-based Approaches.



- Differences in experimental settings:
  - Clone Detection tools and parameter settings.
  - Subject system studied.
  - Programming languages.

- Lack of uniform experimental framework to asses the impacts of clones avoiding possible biases.

Problem Statements

Despite advancements in automated tools. detecting and generating clones, across different programming languages are challenging.

Key Gap:
The full potential of LLMs in **detecting, generating, and managing code clones**—especially across multiple programming language remains **underexplored**.

Opportunity:
Explore how LLMs can address the **longstanding challenges of code cloning** in complex software systems



Summary



WHAT

- Copy/Paste
- Code reuse
- Similar problem-solving approach



WHY

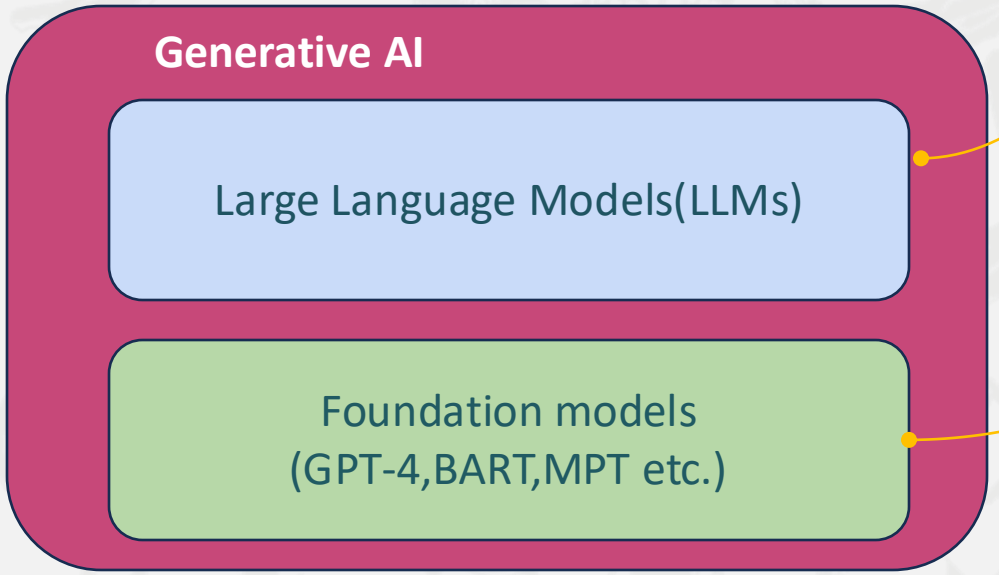
- Saves Time
- Meeting deadlines
- Following established patterns



CHALLENGES

- Maintenance
- Bug fixing
- Code quality

What is LLM?



Large Language Model (LLM):

Model trained on massive datasets to achieve advanced language processing capabilities.

Based on deep learning neural networks.

Foundation Model:

Large ML model trained on vast amount of data & fine-tuned for more specific language understanding and generation tasks.

LLMs are not hype - they change the AI Game

Generative AI & LLMs are once in-a-generation shift in technology

"GPT-4 beats 90% of lawyers trying to pass the bar"

Forbes
03/14/2023

"Falcon is now free of royalties for commercial and research use... Falcon 40B outperforms ... Meta's LLaMA and Stability AI's StableLM"

TII
05/31/2023

"Smaller, more performant models such as LLaMA enable... further democratizing access in this important, fast-changing field..."

Meta AI
02/24/2023

"Vicuna: an open-source chatbot impressing GPT-4 with 90%* ChatGPT quality"



03/30/2023

Why should I care?

ML/AI has been around for a while, why it matters now.

Generative AI models' accuracy and effectiveness have hit a tipping point.

- Powerful enough to enable use cases not feasible even a year ago.
- Economical enough for use even by non-technical users.

Generative AI models' and tooling are readily available.

- Many models are open source and customizable.
- Requires powerful GPUs, but are available in the cloud

"Chegg shares drop more than 40% after company says ChatGPT is killing its business"

05/02/2023

Source: CNBC

"Stack-Overflow is ChatGPT casualty: Traffic Down 14% in March"

19/04/2023

Source: similarweb blog

" 92% of programmers of NA are using AI tools"

13/06/2023

Source: GithubSurvey

Prompt Engineering

There are mainly three kind of prompt engineering techniques:

- Zero Shot Prompting:
  - Asking question directly for generating an answer and handles multiple tasks with a single prompt.
- Few-shot Prompting:
  - Providing a few examples along with the instruction.
- Chain-of-thought:
  - Get the model to work through the steps of the problem.



- Prompt: Give me 10 distinctive implementation for the following code <Code Fragment>
- Prompt: Give me Python, Java, C & C# implementation for the following code <Code Fragments>
- Prompt: Provide only method implementation and start answer with #<targeted programming language>: <SO whole question body>
- Prompt: Give me an accurate but simple translation of the following functions into <Targeted programming language> for code below

LLMs Consideration

Considered state-of-art LLMs along with their popularity:

- **GPT:**
  - Known for high quality natural language understanding and code generation.
  - Excels in generating executable code and solving coding problems.
  - Widely used for coding assistance and chat-based integrations.

- **Llama:**
  - Open source model with different parameter sizes.
  - Optimized for research and applications like coding and NLP tasks.

- **Gemini:**
  - Focused on safe and reliable AI interactions with strong performance in coding.
  - Targets both code quality and cross-language tasks.

LLMs in Clone Generation & Detection





Study-1

Unveiling the potential of LLMs in generating Semantic & Cross-language clones

Study-2

How reliable & usable are the solution codes? A comparative study across Q&A sites, Code Search tools, and Code Generators

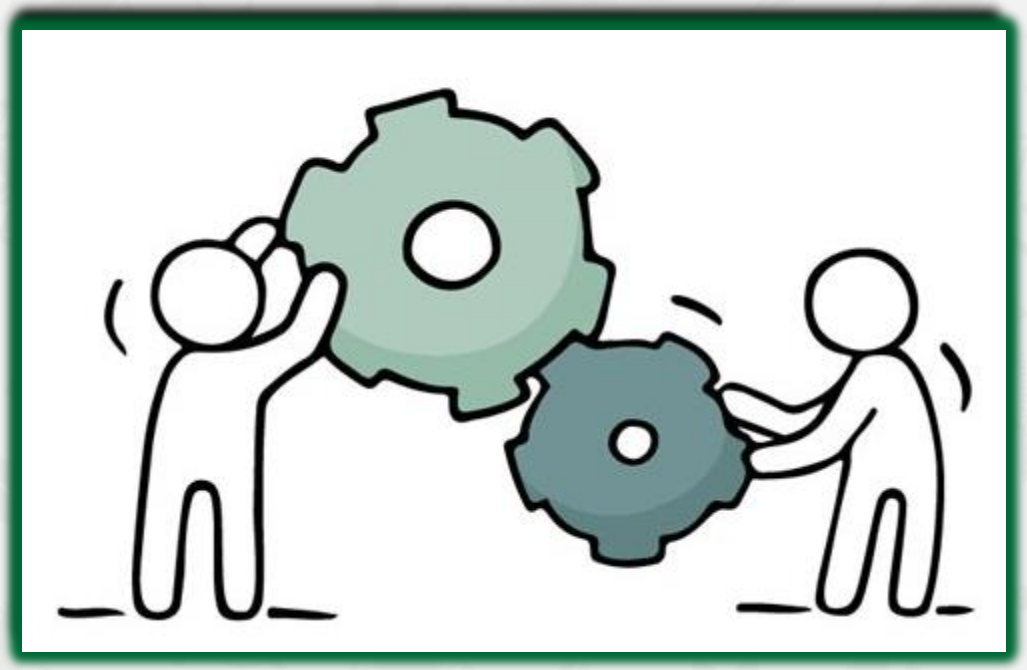
Study-3

From Code Translation to Clone Detection: A Zero Shot Approach

Study-1:Motivation

Unveiling the potential of LLMs in Generating Semantic & Cross-language clones.

- Inconsistent Changes.
- Facilitating Legacy Systems.
- Different Implementation.



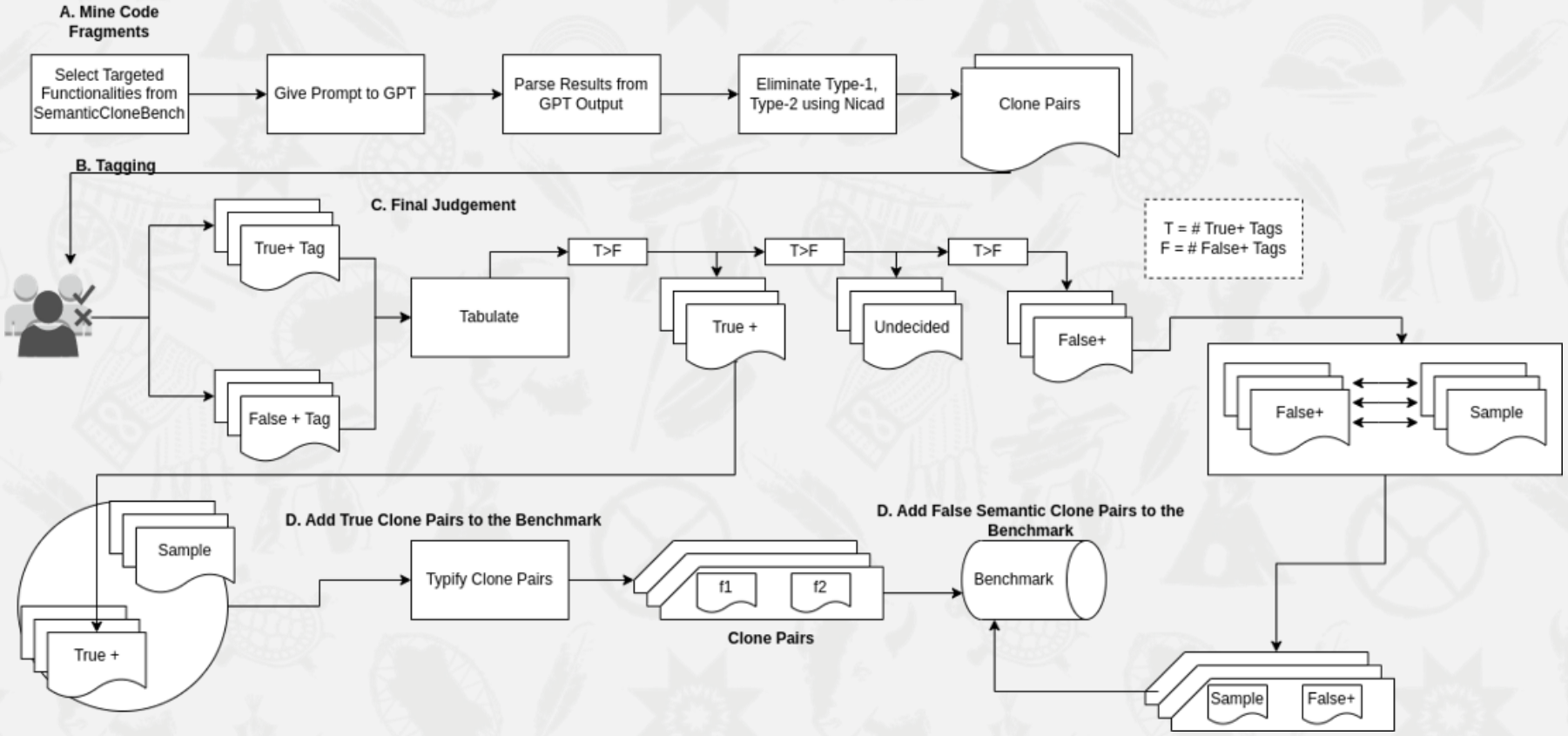
Study-1: Research Questions

- **RQ1:** To what extent does GPT-3 demonstrate the capability to generate high-quality semantic clones?

- **RQ2:** What is the efficiency of GPT-3 to generate cross-language clones (e.g., Java to Python)?



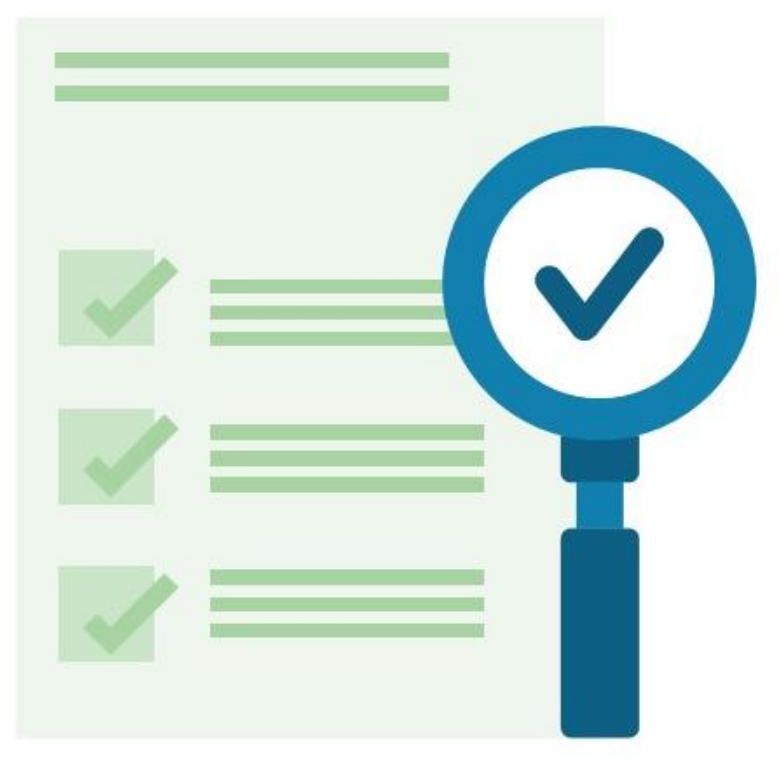
Study-1: Methodology



Study-1: Manual Validation

- Two Rounds and One Discussion Session (For both semantic and cross-language clones)
- Nine Judges (6 undergrad research students and 3 Post-doctoral fellows).





Study-1: Manual Validation(Round-1)

- Divided the semantic clone pairs into three groups after NiCad filtration.
- Each group consists of 2 undergraduate research students.
- Each member of the group validated each code individually.

Clone Pair	Judge 1	Judge 2	Decision
Pair 1	True	True	Selected
Pair 2	False	False	Discarded
Pair 3	Undecided	Undecided	Going to Round 2
Pair 4	True	False	Going to Round 2
Pair 5	False	Undecided	Going to Round 2



Study-1: Manual Validation

- Two Rounds and One Discussion Session (For both semantic and cross-language clones)
- Nine Judges (6 undergrad research students and 3 Post-doctoral fellows).





Study-1: Manual Validation(Round-2)

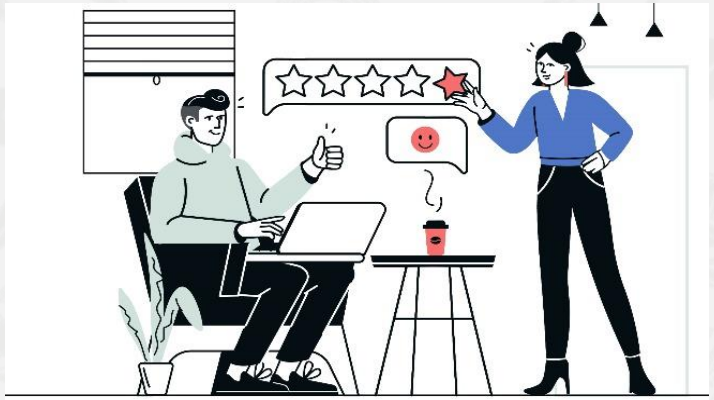
- **Shuffled the undecided or conflicted pairs from Round 1 among the three group.**
- Each group consists of 2 undergraduate research students.
- We have made sure the groups are judging different codes compare to Round 1.

Clone Pair	Judge 1	Judge 2	Decision
Pair 1	True	True	Selected
Pair 2	False	False	Discarded
Pair 3	Undecided	Undecided	Going to Discussion
Pair 4	True	False	Going to Discussion
Pair 5	False	Undecided	Going to Discussion



Study-1:Cohens K Interrater Agreement

- After completing both rounds, we have calculated Cohen Kappa Interrater Agreement among all the groups.



SEMANTIC CLONES COHEN-KAPPA INTERRATER AGREEMENT

Round	Group	Interrater Agreement	Interpretation
Round 1	Group 1	0.700	Substantial Agreement
	Group 2	0.730	Substantial Agreement
	Group 3	0.77	Substantial Agreement
Round 2	Group 1	0.52	Moderate Agreement
	Group 2	0.58	Moderate Agreement
	Group 3	0.54	Moderate Agreement

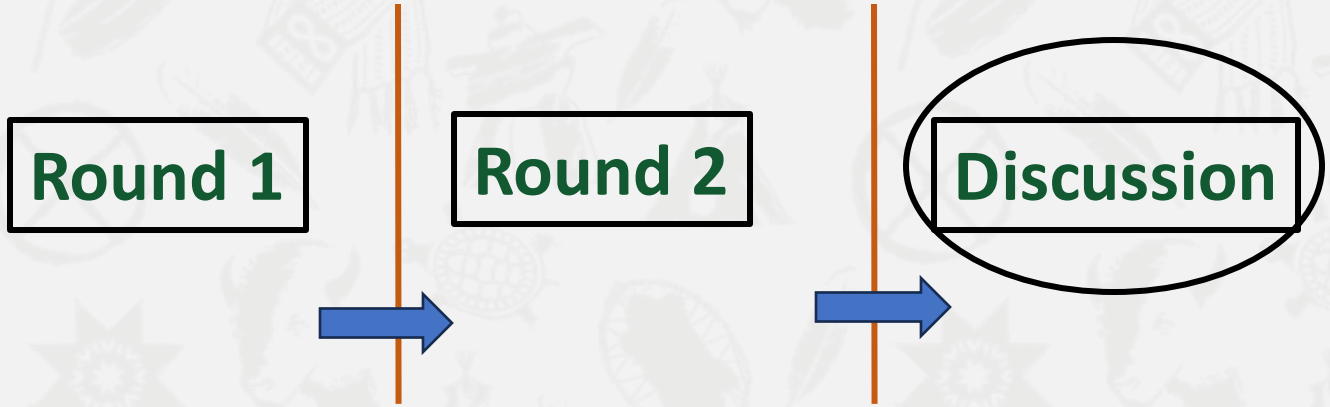
CROSS-LANGUAGE CLONES COHEN-KAPPA INTERRATER AGREEMENT

Round	Group	Interrater Agreement	Interpretation
Round 1	Group 1	0.69	Substantial Agreement
	Group 2	0.59	Moderate Agreement
	Group 3	0.73	Substantial Agreement
Round 2	Group 1	0.48	Moderate Agreement
	Group 2	0.56	Moderate Agreement
	Group 3	0.60	Substantial Agreement

Study-1: Manual Validation

•Two Rounds and One Discussion Session (For both semantic and cross-language clones)

•Nine Judges (6 undergrad research students and 3 Post-doctoral fellows).





Study-1: Discussion on Manual Evaluation

After 2 rounds, if there is still some clone pairs are remaining undecided or conflicted, 3 post-docs discussed among themselves to finally resolve all remaining pairs.

Finally, we have generated 9,321 semantic (from 15,000 clone pairs) and 10,950 cross-language (from 12,000 clone pairs) clone pairs.



Study-1: Discussion

(RQ1) To what extent does GPT-3 demonstrate the capability to generate high-quality semantic clones?

- Showcasing a notable degree of capability in generating high-quality semantic clones, as evidenced by an achieved accuracy of 62.14% with 0.55 BLEU score.
- High BLEU score can be interpreted that GPT generated codes are quality codes.

(RQ2) What is the efficacy of GPT-3 in accurately converting code snippets from one programming language to another?

- Demonstrating ability to comprehend the structural and syntactical intricacies inherent to different programming languages with accuracy of 91.25%.

Threats to validity

- Real vs Artificial Clone
- Biasness in Manual Evaluation



Summary

•This research aims to introduce a transformative paradigm for code reuse, refactoring, migration and renovation.

•After thorough validation process, we have managed to get 9,321 true semantic clone pairs and 10,950 cross-language clone pairs after handling all the limitations of GPT-3.

•With a noteworthy accuracy rate of 62.14% and a 0.55 BLEU score, GPT-3 showcases its potential in accurately replicating semantic structures of a code.

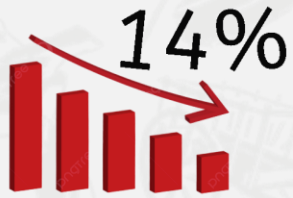
•With an accuracy of 91.25% for cross-language clones, GPT shows prowess on generating cross-language clones.

Study-2: Motivation

How reliable & usable are the solution codes? A comparative study across Q&A sites, Code Search tools, and Code Generators

Find out which tool we should prefer:

- Usability.
- Accuracy.
- Code Quality.



SO traffic down after GPT release





Code is generated by AI in **GOOGLE**
-Sundar Pichai, CEO of Google

92%

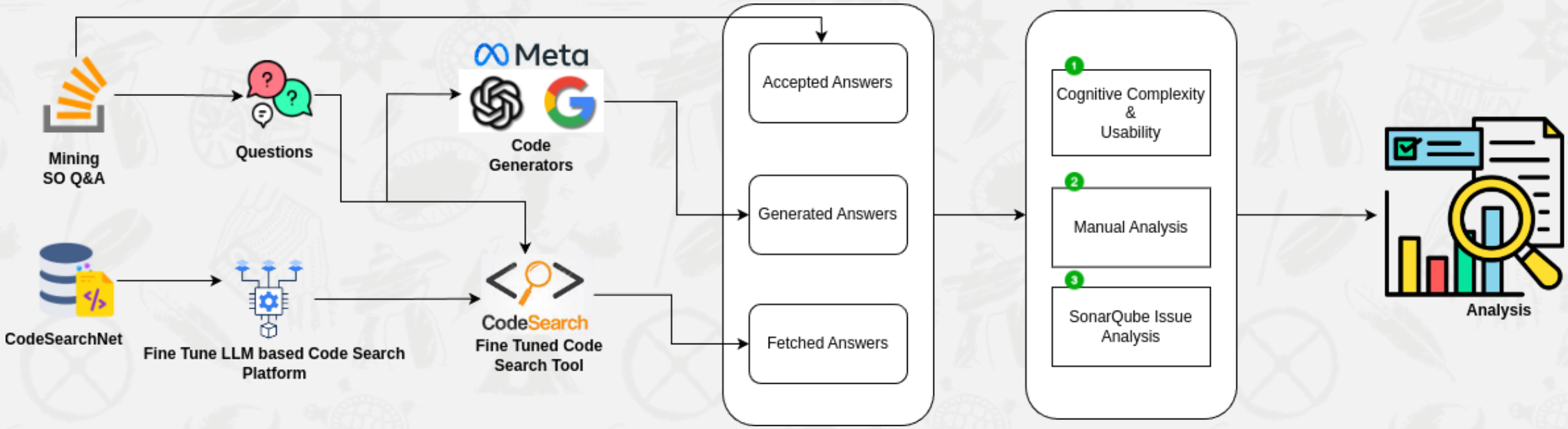
North American developers are using AI-assisted tools

Study-2: Research Questions

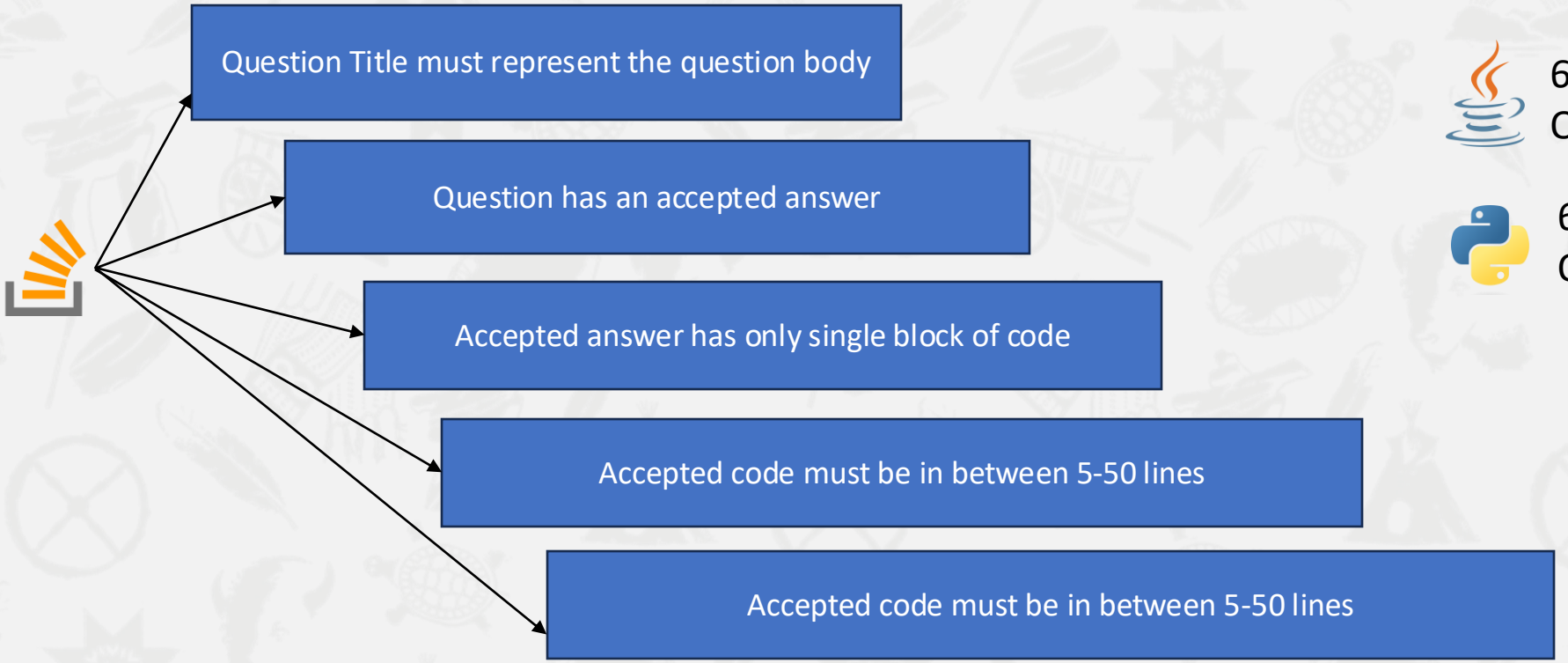
- **RQ1:** To what extent are the code snippets from stack overflow, gpt-4, gemini, llama, CodeBERT, and GraphCodeBERT usable (compilable/runnable)?
- **RQ2:** Are the solutions provided by Stack Overflow, GPT, Gemini, Llama, CodeBERT, and GraphCodeBERT accurate?
- **RQ3:** What quality threats, such as bugs, code smells, and vulnerabilities, are present in the solution code, and what is their severity?



Study-2: Methodology



Study-2: SO Question Selection



6,776
Question-Answer pairs

6,032
Question-Answer pairs

Study-2: Mitigating Selection Bias

- Accepted answer is not always upvoted/liked response.
- Manual Evaluation was conducted.

How to remove string between unbalanced brackets in Python3 without Regex

```
def remove_unbalance(s):
    last_open = s.rfind("(")
    last_close = s.rfind(")")
    return s[:last_open] if last_open > last_close else s
```

- Efficient use of built in methods.
- Simplicity.
- Handling edge cases.

Study-2: Code Search Tools

- Removed Comments in the code.
- Filtered out external resources like “website link”.

PL	Training	Dev	Test	Candidate Code
Python	251,820	13,914	14,918	43,827
Java	164,923	5,183	10,955	40,347

CodeSearchNet Dataset for Fine Tuning

PL	Python	Java
CodeBERT	0.75	0.71
Graph	0.7	0.72

MRR Results

Study-2: Findings (RQ1)

Source/Tool	Executable
Stack Overflow	76.7
CodeBERT	74.5
GraphCodeBERT	63.0
GPT-4	87.7
Gemini	85.7
LlaMA-3-8B	83.2

Python Executable Codes

Source/Tool	Compilable
Stack Overflow	79.3
CodeBERT	45.0
GraphCodeBERT	58.0
GPT-4	82.7
Gemini	75.5
LlaMA-3-8B	74.3

Java Compilable Codes

Study-2: Findings (RQ1)

Tools	Median	Mean
SO	2.0	3.14
CodeBERT	3.0	4.49
GraphCodeBERT	1	1.98
GPT-4	1	2.19
Gemini	1	2.27
LLama	3	3.75

Cognitive Complexity for Python

Tools	Median	Mean
SO	1	2.12
CodeBERT	0	0.80
GraphCodeBERT	0	0.31
GPT-4	1	2.88
Gemini	1	1.93
LLama	2	3.89

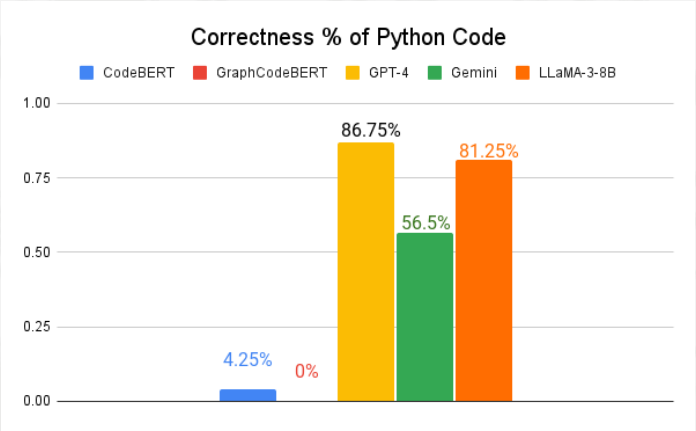
Cognitive Complexity for Java

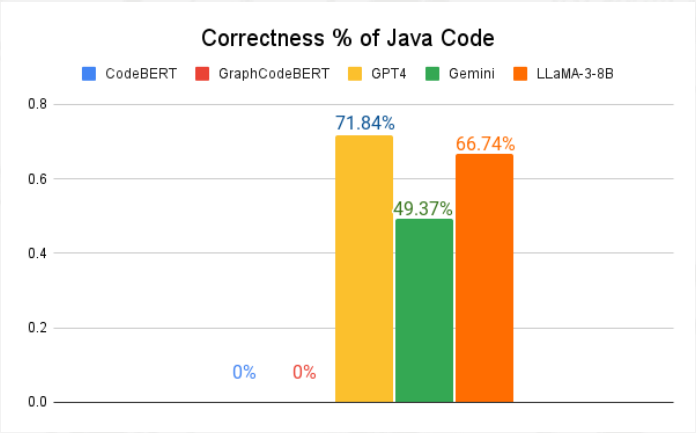
Study-2: Findings (RQ2)

Manual Evaluation

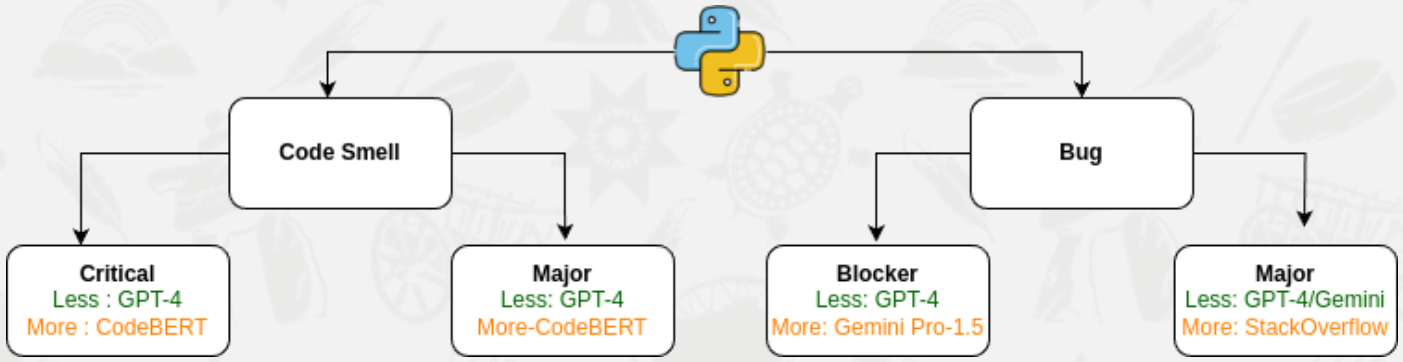
- Used SO answers as reference.
- Functional testing.
- 2 Judge.
- 53-Person hours.

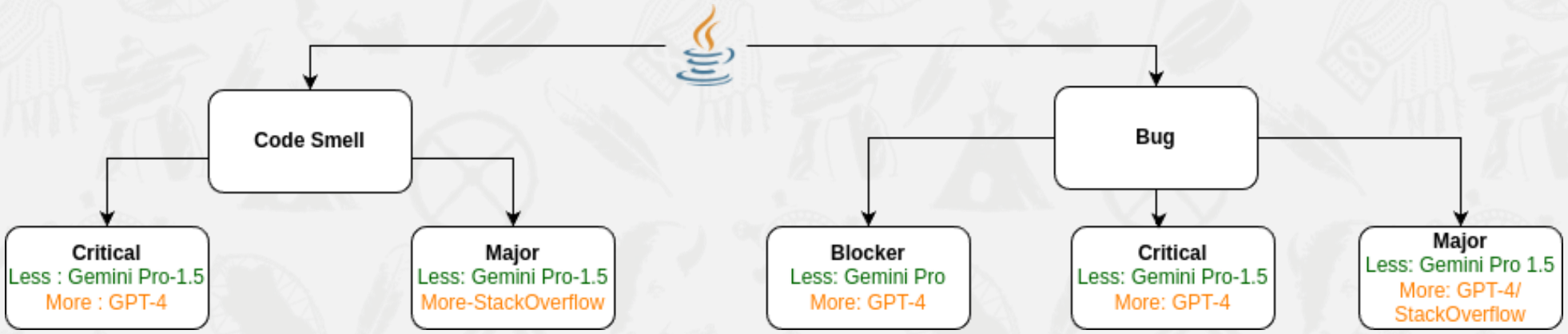
Interrater Agreement	Interpretation
0.91	Almost Perfect Agreement



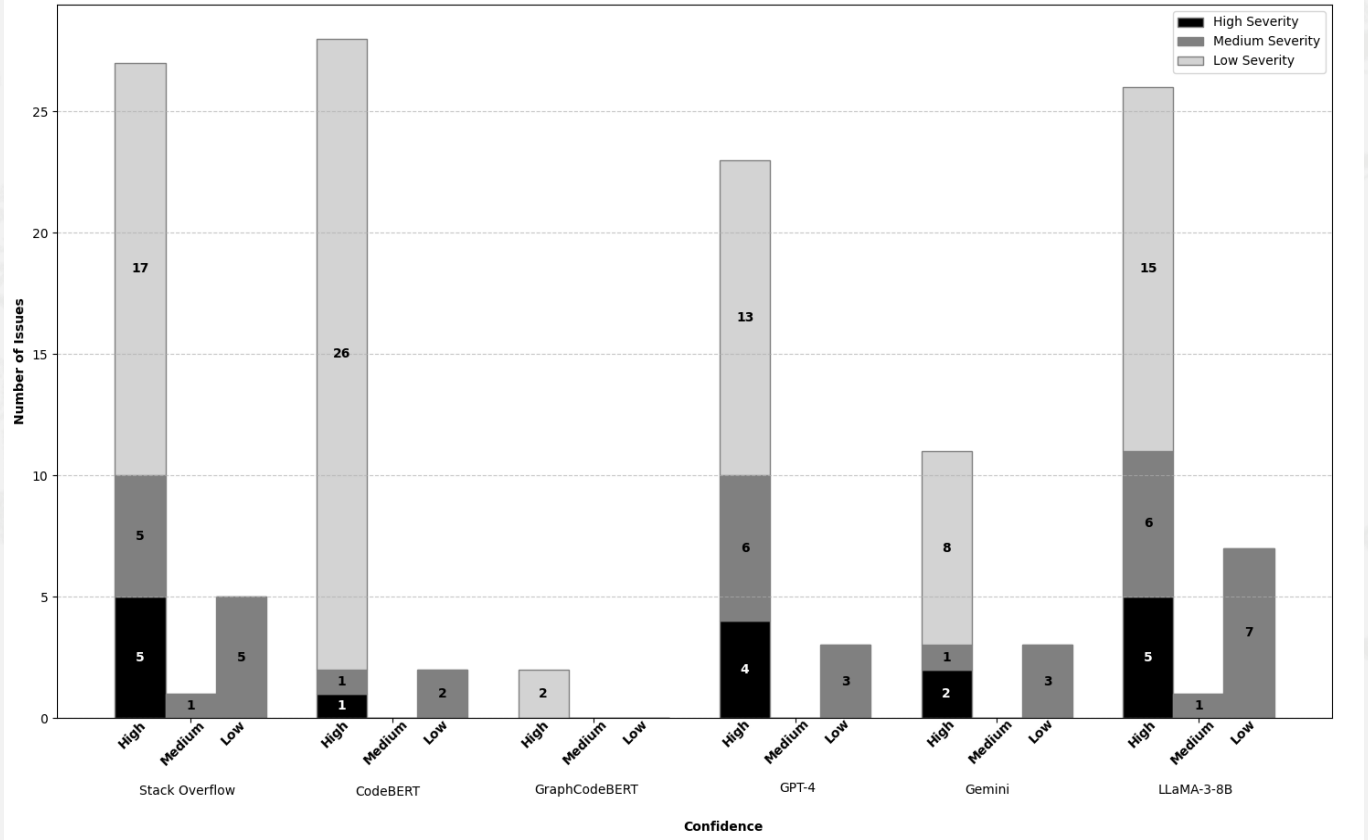


Study-2: Findings(RQ3)



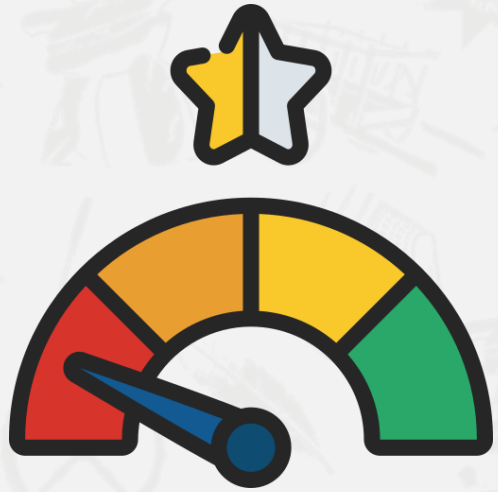


Study-2: Findings(RQ3)-cross validation



Study-2: Bad performance of Code Search tools

- Can not work on query contains more than 512 tokens.
- Code search tools prioritize function names over the body of the code when discerning the meaning of the code, emphasizing the importance of function names for accurate retrieval.
- Obfuscating function names leads to a noticeable drop in performance.
- Comments do not have any influence as it were removed while fine-tuning



Study-2: Summary

- This study aims to Identify accurate and reliable sources of LLM for effective coding solutions.
- GPT outperforms other tools/platforms with 80% of the code examples being executable/compilable and produces more readable code in python programming langauges.
- Gemini excels in readability for java code examples.
- Code search tools performs very poor in the similar situation.

Study-3: Motivation

From Code Translation to Clone Detection: A Zero-Shot Cross-Language Approach

- Easier refactoring.
- Increase quality.
- Better Maintenance.
- Decrease complexity in multi platform environments.
- Overall better clone management.

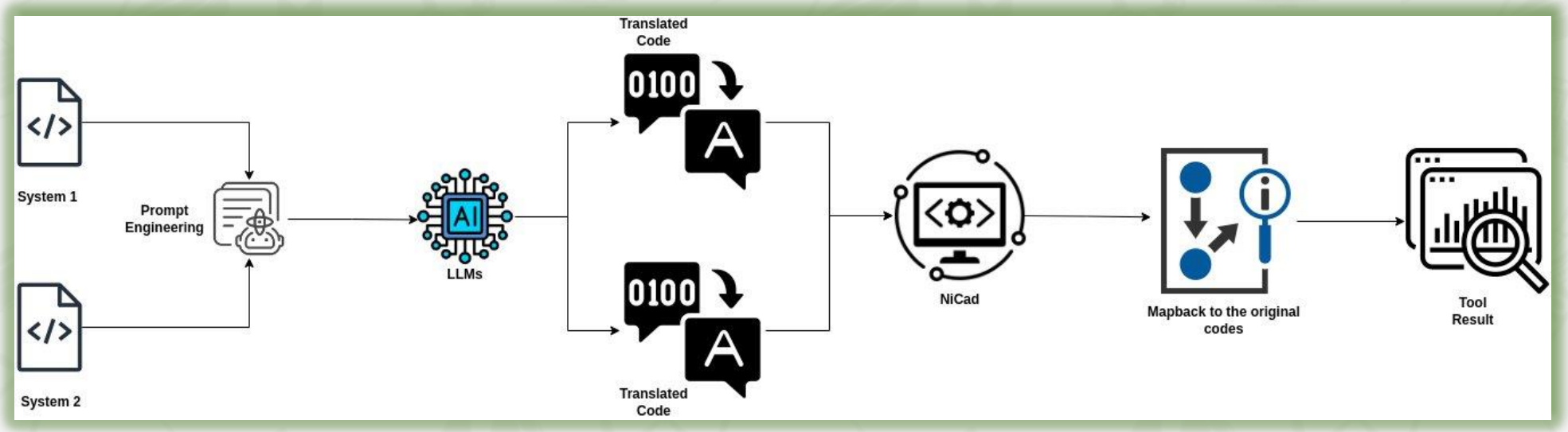


Study-3: Research Questions

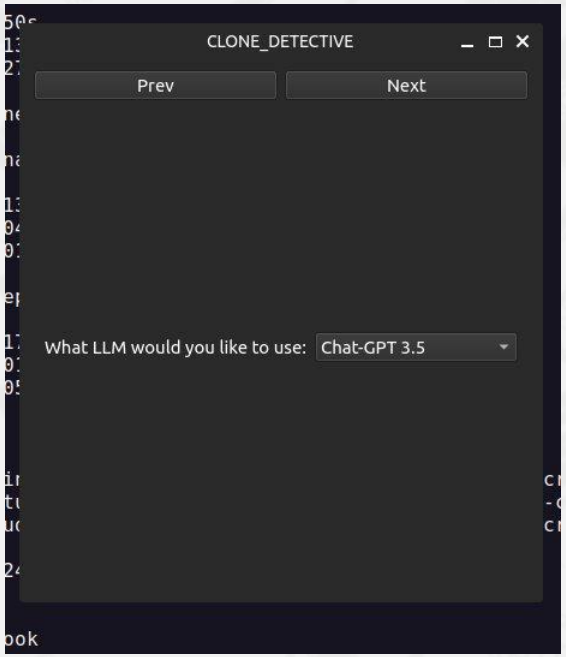
- **RQ1:** How effective is the proposed method or tool(Lang2Clone) in detecting cross-language code clones?
- **RQ2:** How does the performance of the proposed tool vary across different programming language pairs?
- **RQ3:** Does translating code to a third language impact clone detection performance?

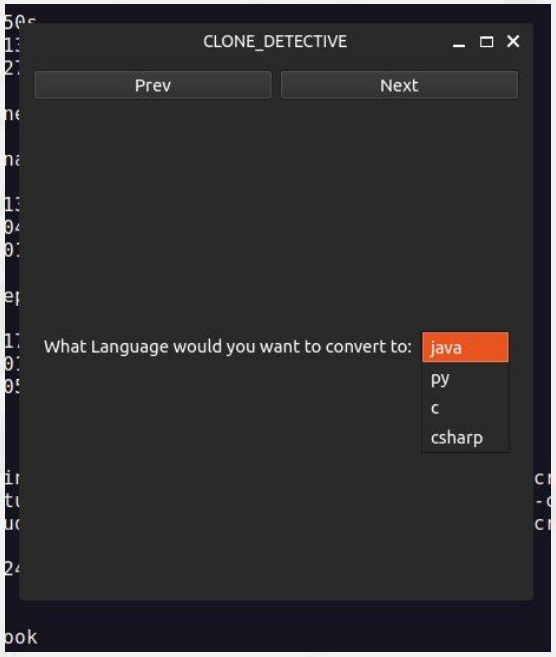


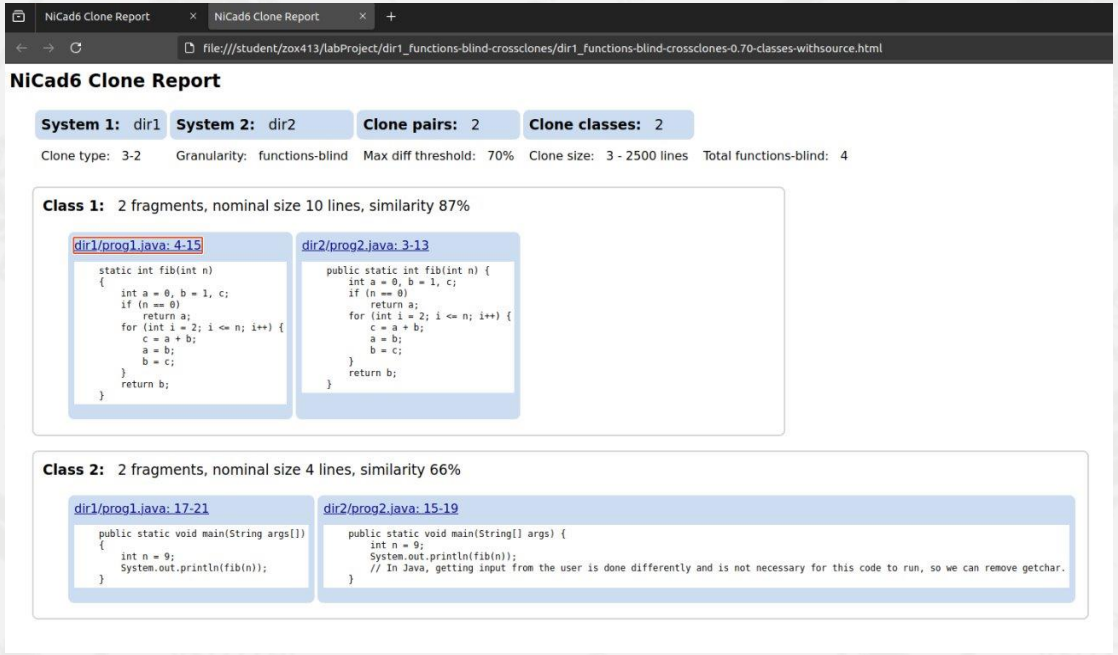
Study-3: Methodology



Study-3: Tool Overview





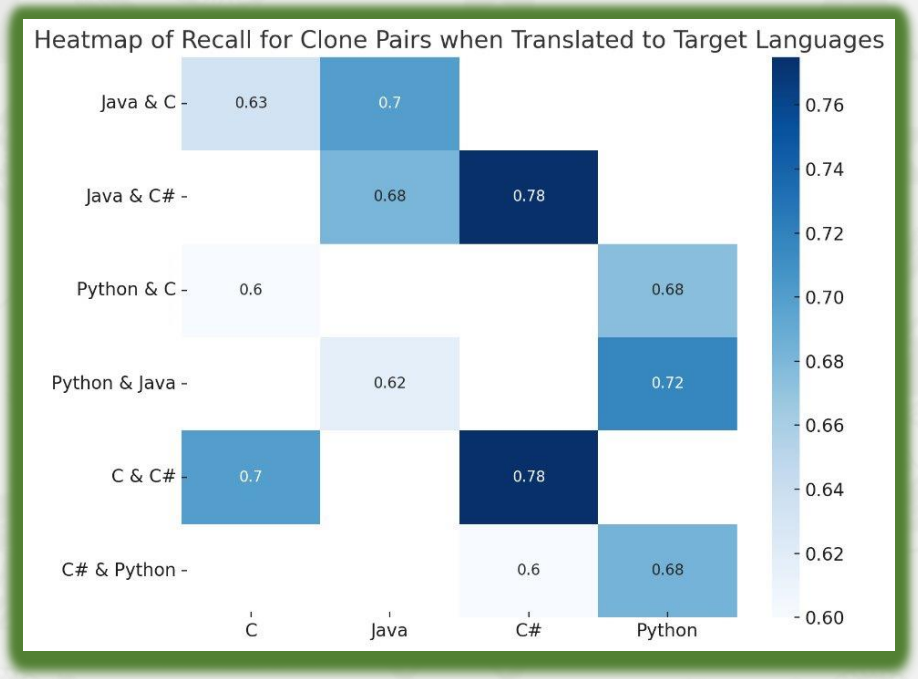


Study-3: Results Comparisons

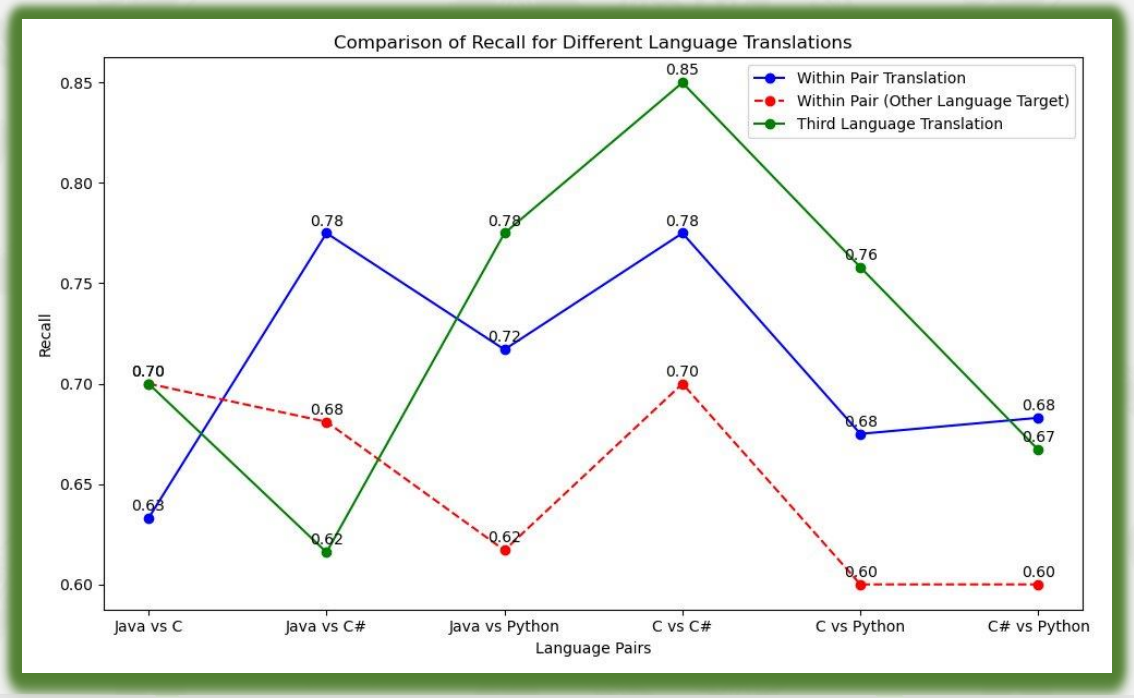
Tools	LeetCode Recall	GPTCloneBench Precision	GPTCloneBench Recall	GPTCloneBench F1
Lang2Clone	0.82	0.92	0.71	0.81
GraphCodeBERT	-	0.1	0.75	0.176
CLCDSA	-	0.84	0.1	0.178

Study-3: Translation within language

Language Pairs	Target Language	Recall
Java vs C	C	0.633
Java vs C	Java	0.700
Java vs C#	C#	0.775
Java vs C#	Java	0.681
Java vs Py	Py	0.717
Java vs Py	Java	0.617
C vs C#	C#	0.775
C vs C#	C	0.700
C vs Py	Py	0.675
C vs Py	C	0.600
C# vs Py	Py	0.683
C# vs Py	C#	0.600



Study-3: Translation to third language



Language Pairs	Target Language	Recall
Java vs C	C#	0.700
Java vs C	Python	0.691
Java vs C#	C	0.540
Java vs C#	Python	0.616
Java vs Python	C	0.525
Java vs Python	C#	0.775
C vs C#	Java	0.850
C vs C#	Python	0.758
C vs Python	Java	0.750
C vs Python	C#	0.650
C# vs Python	Java	0.667
C# vs Python	C	0.430

Study-3: Threats

- Bias in Programming language selection.
- Quality of prompts.
- Tie with NiCad.
- Dataset may not fully represent diversity of real-world scenarios.



Conclusion

- LLMs can generate semantic and cross-language clone.
- GPT is one of the most prominent LLM compared to other LLMs available in most cases in terms of code quality.
- Utilizing Code translation we can detect cross-language clones.



Publications/Submitted/Ready to submit

1. Roy, Palash R., et al. "Unveiling the potential of large language models in generating semantic and cross-language clones." 2023 IEEE 17th International Workshop on Software Clones (IWSC). IEEE, 2023. (**Peoples Choice Best Paper Award, Student Encouragement Award**)
2. Alam, Ajmain I., Palash R. Roy et al. "GPTCloneBench: A comprehensive benchmark of semantic clones and cross-language clones using GPT-3 model and SemanticCloneBench." 2023 IEEE International Conference on Software Maintenance and Evolution (ICSME). IEEE, 2023. (**21% Acceptance rate**)
3. Palash R. Roy, et al. "How reliable and usable are the solution codes? A comparative study across Q&A Sites, Code Search Tools, and Code Generators." presented at 8th ICSAC Research Fest(**Best 3MT Presentation & Poster Award**)(To be submitted to EMSE Journal).
4. Roy, Palash R., et al. "From Code Translation to Clone Detection: A Zero Shot Cross-Language Approach. "(To be Submitted at International Mining Software Repositories (MSR)
5. Billah, Mustakim., Roy, Palash, et al. "Are Large Language Models a Threat to Programming Platforms? An Exploratory Study" 2024 International Symposium on Empirical Software Engineering and Measurement. (**20% acceptance rate**)
6. Roy, Palash, et al. "Exploring Usability Frontiers: Unleashing semantic and cross-language clones with LLMs", presented at 2023 Software Analytics and Research Symposium 2023. (**Peoples Choice Best Presentation Award**)
7. Alam, Ajmain, Roy, Palash et al. "Does Code Clone Detection really need all those layers?" (To be Submitted at International Conference on Automated Software engineering)

Acknowledgement

This thesis is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grants program, and by the industry-stream NSERC CREATE in Software Analytics Research (SOAR).

Acknowledgements are also due to the Director of SOAR, Prof. Chanchal K. Roy for his guidance, help and time in meeting all the different requirements of the industry-stream SOAR program by arranging lectures in addition to the formal courses, professional and technical skill requirements by arranging professional development workshops, talks, Industry and Mitacs workshops, graduate professional skills development programs, entrepreneurship training, and industrial internships.

References

- Krinke, J. (2007, October). A study of consistent and inconsistent changes to code clones. In *14th working conference on reverse engineering (WCRE 2007)* (pp. 170-178). IEEE.
- Krinke, J. (2008, September). Is cloned code more stable than non-cloned code?. In *2008 Eighth IEEE International Working Conference on Source Code Analysis and Manipulation* (pp. 57-66). IEEE.
- Krinke, J. (2011, May). Is cloned code older than non-cloned code?. In *Proceedings of the 5th International Workshop on Software Clones* (pp. 28-33).
- Alam, A. I., Roy, P. R., Al-Omari, F., Roy, C. K., Roy, B., & Schneider, K. A. (2023, October). GPTCloneBench: A comprehensive benchmark of semantic clones and cross-language clones using GPT-3 model and SemanticCloneBench. In *2023 IEEE International Conference on Software Maintenance and Evolution (ICSME)* (pp. 1-13). IEEE.
- Roy, P. R., Alam, A. I., Al-Omari, F., Roy, B., Roy, C. K., & Schneider, K. A. (2023, October). Unveiling the potential of large language models in generating semantic and cross-language clones. In *2023 IEEE 17th International Workshop on Software Clones (IWSC)* (pp. 22-28). IEEE.
- Mondal, M., Roy, B., Roy, C. K., & Schneider, K. A. (2020, February). HistoRank: History-based ranking of co-change candidates. In *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER)* (pp. 240-250). IEEE.
- Lozano, A., & Wermelinger, M. (2008, September). Assessing the effect of clones on changeability. In *2008 IEEE International Conference on Software Maintenance* (pp. 227-236). IEEE.
- Lozano, A., & Wermelinger, M. (2010, May). Tracking clones' imprint. In *Proceedings of the 4th International Workshop on Software Clones* (pp. 65-72).
- Juergens, E., Deissenboeck, F., Hummel, B., & Wagner, S. (2009, May). Do code clones matter?. In *2009 IEEE 31st International Conference on Software Engineering* (pp. 485-495). IEEE.
- Barbour, L., Khomh, F., & Zou, Y. (2011, September). Late propagation in software clones. In *2011 27th IEEE International Conference on Software Maintenance (ICSM)* (pp. 273-282). IEEE.
- Mondal, M., Roy, C. K., & Schneider, K. A. (2018). Bug-proneness and late propagation tendency of code clones: A comparative study on different clone types. *Journal of Systems and Software*, *144*, 41-59.
- Roy, C. K., Zibran, M. F., & Koschke, R. (2014, February). The vision of software clone management: Past, present, and future (keynote paper). In *2014 Software Evolution Week-IEEE Conference on Software Maintenance, Reengineering, and Reverse Engineering (CSMR-WCRE)* (pp. 18-33). IEEE.

Thank You



Questions ?

Key Take Away

This thesis explores the power of LLMs in code generation and clone detection. It reveals LLMs capability of generating reusable codes, compares code quality from LLMs and human sources and introduces Lang2Clone- a hybrid tool combining LLMs translation with Nicad's detection, advancing multilingual code analysis and AI driven software development.

