Designing(Creating) a Research Poster

Clear, and worth stopping for

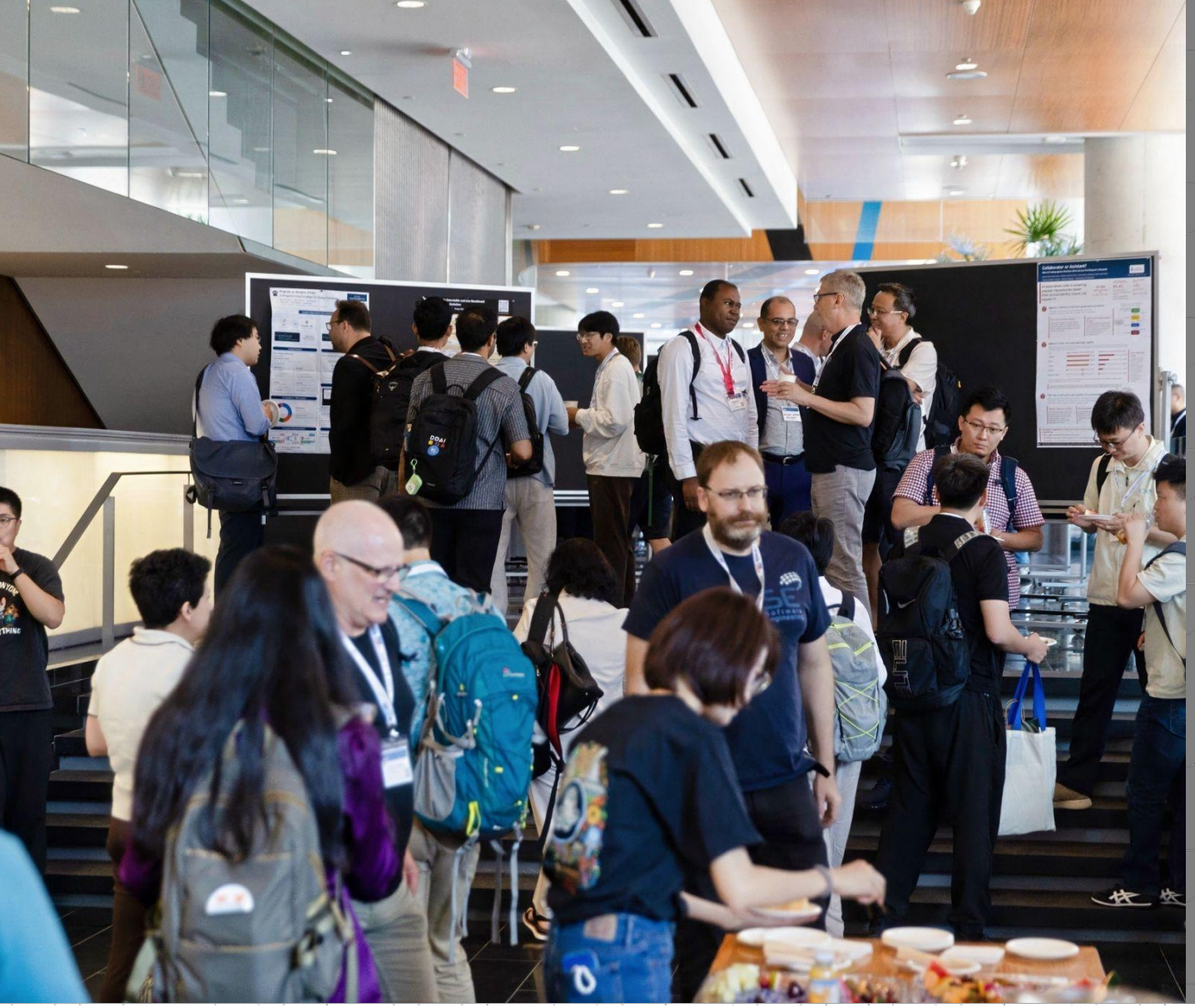
Palash R. Roy
PhD Student, Department of Computer Science
Grad Peer Mentor, USask Library

Sept 24, 2026

TRANSITION TO GRAD STUDIES WORKSHOP

UNIVERSITY OF SASKATCHEWAN
University Library
LIBRARY.USASK.CA

CC BY NC SA



Most posters are a wall of text nobody reads.

A good poster earns a thirty-second stop in a busy, tired hall.

CC BY NC SA

By the end, you will be able to

- Build a poster around one clear message
- Lay it out so the eye follows an easy path
- Design it to be readable and figure-led
- Present it with a short pitch and handle questions





POLL

- Have you made a research poster before?
- Which part worries you most, content, layout, design, or presenting?



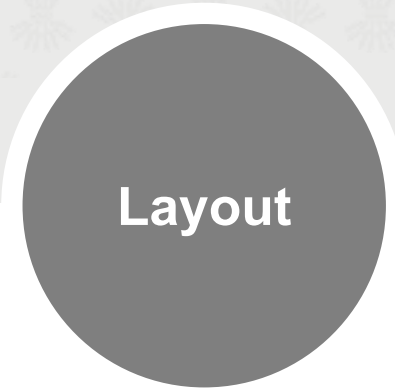
CC BY NC SA

Four things make a poster work



One clear point

Say one thing, well



An easy path

Guide the eye

Readable, visual

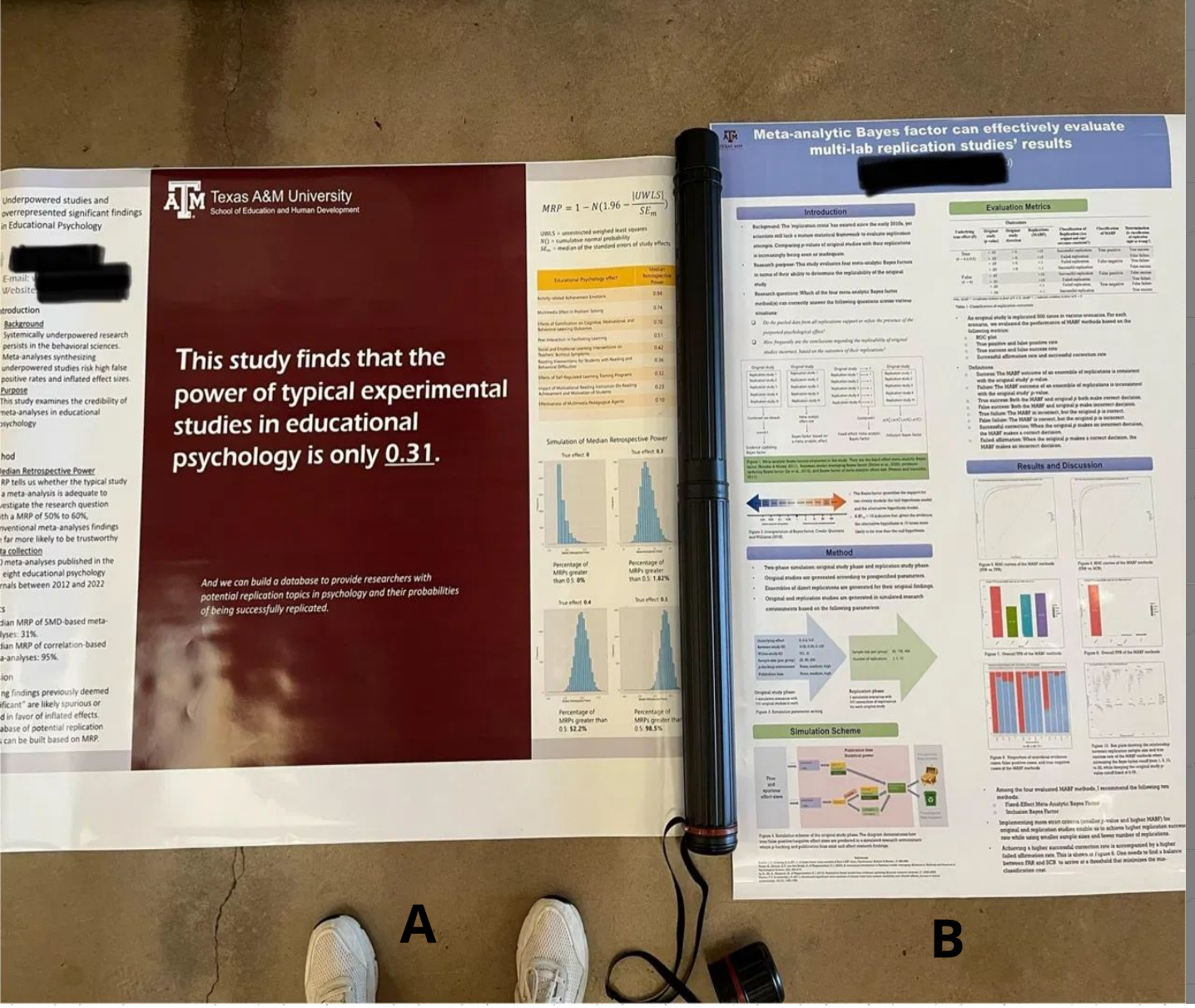
Big text, strong figures



A confident pitch

Sixty seconds, then talk

CC BY NC SA



Which poster would make you stop and walk over?

CC BY NC SA

1

Start with One Message



A poster is a billboard, not a paper

- One main finding, not your whole thesis
- If they read nothing else, what should they take away
- Cut text hard, aim for far less than feels safe

CC BY NC SA

Put your finding in the title

- A title that states the result, not just the topic
- A short takeaway line people can read from across the room
- Keep sections for context, method, results, and meaning



2

Layout Guides the Eye



Give the reader one clear path

- Use columns that flow top to bottom, then left to right
- Number your sections so the order is obvious
- Do not make people hunt for where to start

CC BY NC SA

Use hierarchy and white space

Big headings, smaller body, clear sections

Leave empty space so the poster can breathe

Group related things, separate unrelated things



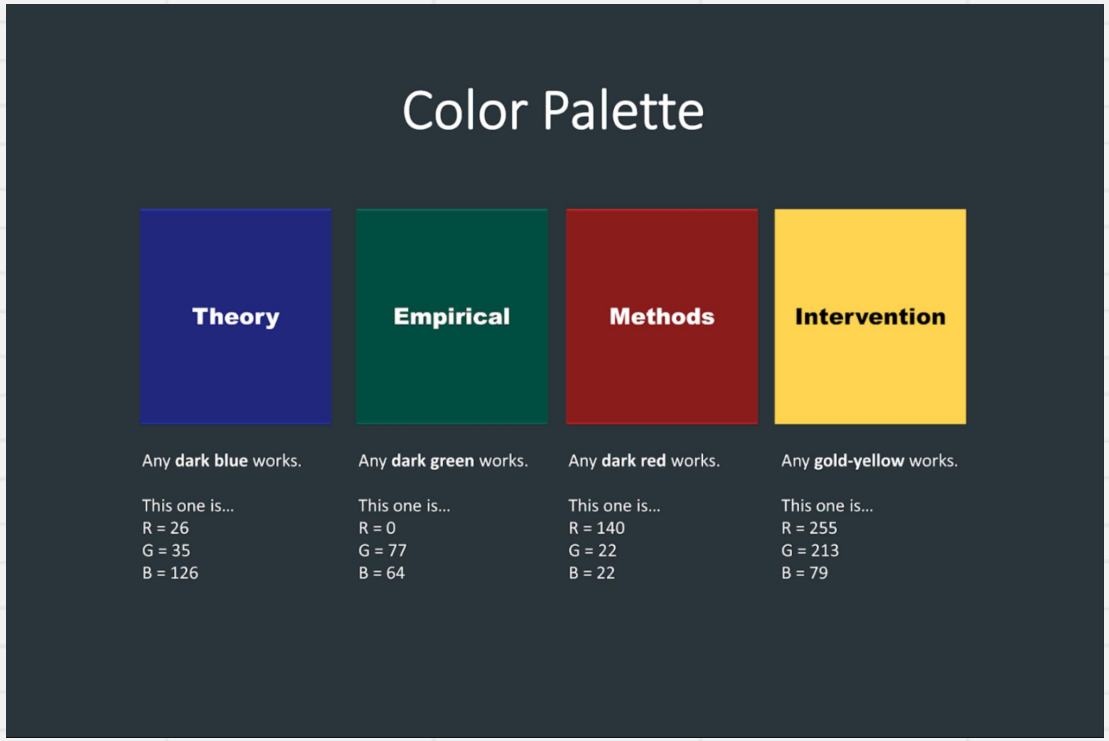
3

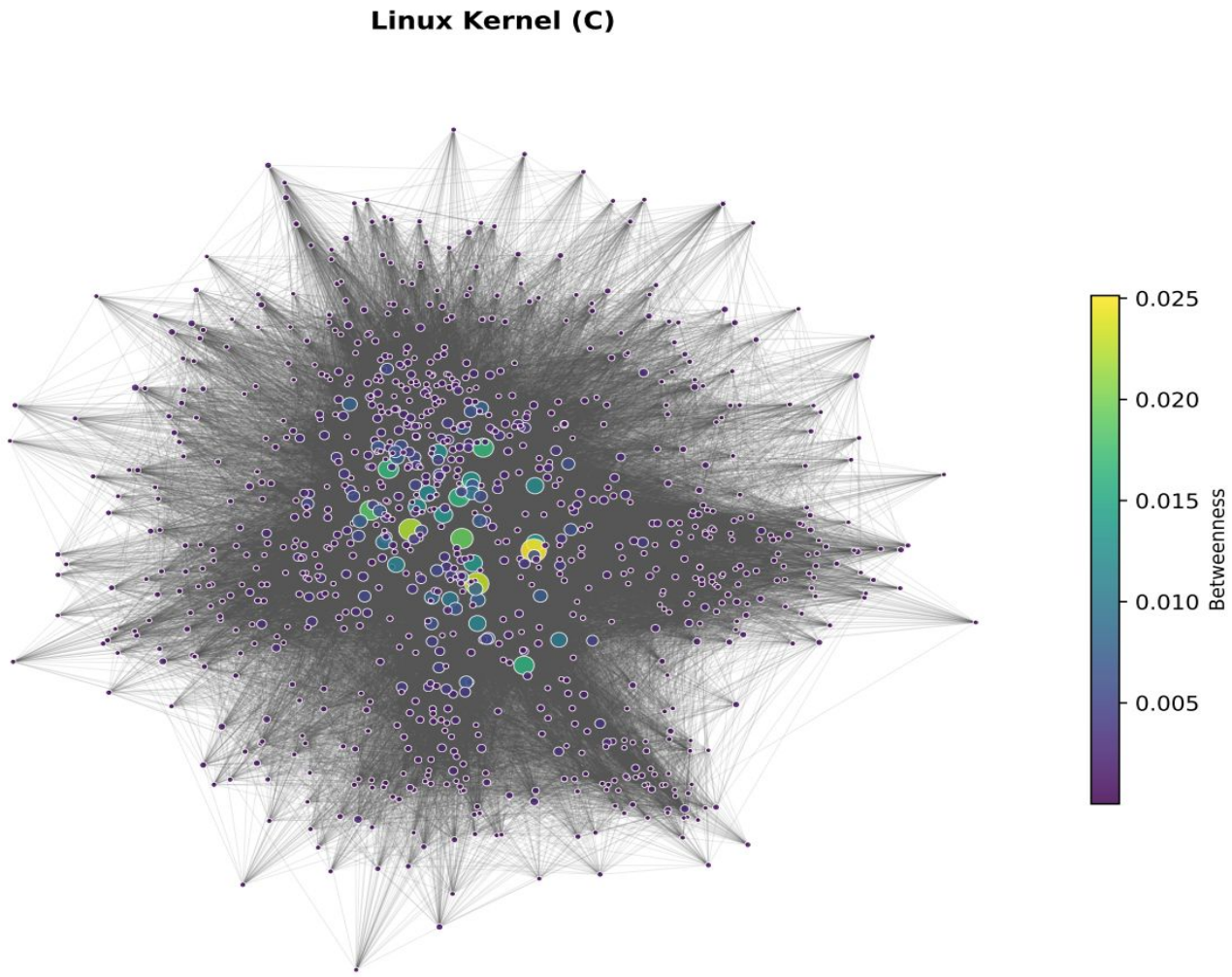
Design for Clarity

Big fonts, few colors, strong figures

- Readable from about two metres away
- Two or three colors, used with purpose
- One figure that tells the story on its own

visit : https://coolors.co/





Let figures do the talking

- Replace paragraphs with a chart, a diagram, or a photo
- Label figures so they stand alone
- A clear figure beats a column of text

CC BY NC SA

4

Present with Confidence

Have a sixty-second pitch ready

- Open with the problem and why it matters
- Give your one finding, then how you got it
- End with what it means, and invite questions



CC BY NC SA

Handle questions and engage

- Invite people in, do not wait to be approached
- Listen fully, then answer the question asked
- If you do not know, say so, and offer to follow up



CC BY NC SA

From Code Translation to Clone Detection: A Zero Shot Approach

Palash R. Roy, Ajmain I. Alam, Banani Roy, Kevin Schneider, Chanchal Roy

Acknowledgement

This work is supported by NSERC Discovery Grants, the CFI John R. Evans Leaders Fund, and the industry-stream NSERC CREATE SOAR program.

Affiliations

University of Saskatchewan · SOAR · NSERC CRSNG · Software Research Lab · iSE LAB Interactive Software Engineering and Analytics Lab

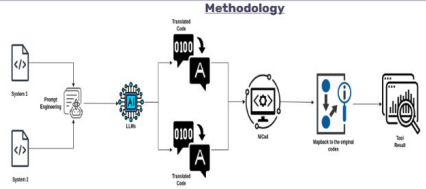
Introduction

Background: Code clones are similar code fragments created through reuse [2]. They speed up development but add redundancy and maintenance effort. Clones typically account for 9 to 17 percent of a system's codebase and can reach as high as 50 percent [1]. In the modern AI era, repositories contain up to four times more clones.

Problem: Cross-language development makes clone management difficult. Existing tools work well within a single language [3] but struggle across languages like Java, C sharp, Python, Ruby, or Rust. Many rely on revision histories, support only limited languages, or fail to scale [7]. With AI-assisted coding, multi-language systems are growing rapidly, and developers often migrate code from one language to another [6], creating cross-language clones that hold important architectural knowledge yet remain underexplored [4].

Contribution: This study proposes **Lang2Clone**, which uses large language models to translate code across languages before applying a syntactic clone detector, enabling more flexible and accurate detection in multi-language environments.

Methodology



Related Work

CLCDSA [4] uses labeled cross-language clone datasets to train a supervised detector, but its coverage is limited by the availability of parallel data and its low recall in diverse language pairs. CodeBERT [5], a multilingual pre-trained model, enables zero-shot clone detection, yet it often relies on lexical similarity and struggles with deeper semantic alignment across languages. Compared to these approaches, Lang2Clone uses LLM-based translation combined with syntactic detection, providing a more flexible way to handle structural and semantic variation across languages.

Research Questions

RQ1: How effective is our tool Lang2Clone for cross-language clone detection compared to existing tools?

RQ2: How does performance vary across different language pairs?

RQ3: Does translating both languages into a third language change clone detection accuracy?

Results

Lang2Clone delivers the strongest overall performance. On GPTCloneBench, it achieves the highest precision (96%), solid recall (71%), and the best F1 score (81%). By comparison, GraphCodeBERT has high recall (75%) but very low precision (10%), and CLCDSA has high precision (84%) but very low recall (10%), resulting in weak F1 scores for both.

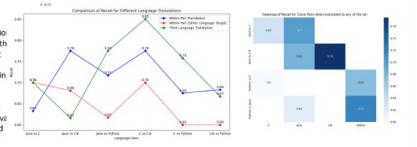
Across language-pair evaluations, Lang2Clone performs best when translating between structurally similar languages. Recall is highest when targeting Java, C sharp, or Python, while translations into C show the weakest results. In third-language experiments, Java and C sharp again yield the strongest performance, reaching a maximum recall of 0.850.

Analysis

RQ1: Effectiveness of Lang2Clone
Lang2Clone provides a balanced and reliable approach to cross-language clone detection. Its high precisio shows that its translations produce clean, comparable code, allowing NiCad to distinguish true clones with minimal noise. The recall indicates that the method captures a broad range of clone variations, making it more practical than tools that overpredict (like GraphCodeBERT) or underpredict (like CLCDSA). This balance highlights the advantage of combining code translation with syntactic detection rather than relyin on embeddings alone.

RQ2: Variation Across Language Pairs
Performance differences across language pairs align closely with language paradigms. Lang2Clone works best when languages share structural features—particularly statically typed, object-oriented pairs like Java and C sharp. Translation becomes harder when moving between object-oriented, dynamically typed, and procedural languages, which reduces the similarity NiCad can capture. This suggests that paradigm compatibility plays a larger role than syntax alone in successful cross-language clone detection.

RQ3: Effect of Translating to a Third Language
Translating into a third language can either improve or reduce performance depending on how well that target language aligns with the paradigms of the original pair. Java and C sharp often enhance recall because they provide consistent, well-structured representations. In contrast, C yields lower recall due to its procedural nature and limited support for higher-level constructs. These patterns show that the choice of target language influences clone detectability and that structured, statically typed languages serve as more effective "bridges" for cross-language clone analysis.



Tool	LeetCode	GPTCloneBench		
	Recall	Precision	Recall	F1 Score
Lang2Clone	0.82	0.96	0.71	0.81
GraphCodeBERT	-	0.1	0.75	0.176
CLCDSA	-	0.84	0.1	0.178

Table 5.1: Results Comparison of Tools

Threats to validity

Language Selection
The study uses java, c, C# and python. While this may limit generalizability to newer or less common languages, our intention was to cover both object oriented, dynamic or scripting language

LLM Translation Validity
Translation accuracy depends on prompt quality and training data. We mitigated this by testing multiple prompts and selecting most reliable one.

Conclusion

Lang2Clone shows strong and balanced performance for cross-language clone detection, outperforming GraphCodeBERT and CLCDSA by achieving high precision with competitive recall. It performs best when languages share similar paradigms, such as Java and C sharp, and recall decreases when translating across very different paradigms, particularly when targeting procedural C. Translating into a third language can help or hinder accuracy depending on how well the target language preserves structure. Overall, Lang2Clone offers an effective and flexible approach for detecting clones in modern multi-language software systems.

References

[1] Alam, A. I., Roy, P. R., Al-Omari, F., Roy, B., Roy, C. K., & Schneider, K. A. (2023). *GPTCloneBench: Benchmarking semantic and cross-language clones.* In ICSME.
[2] Roy, C. K., & Cordy, J. R. (2007). *A survey on software clone detection research.* Queen's Univ. Tech. Report.
[3] Mandal, M., Roy, C. K., & Schneider, K. A. (2014). *Ranking clones for refactoring using association rules.* In CSMR-WCRE.
[4] Nafi, K. W., Kar, T. S., Roy, B., Roy, C. K., & Schneider, K. A. (2019). *CLCDSA: Cross-language clone detection using syntactic features and API docs.* In ASE.
[5] Arshad, S., Abid, S., & Shamail, S. (2022). *CodeBERT for code clone detection: A replication study.* In IWSC.
[6] GitClear (2025). *AI Copilot Code Quality: A 12-month analysis.* GitClear Report.
[7] Roy, P. R., Alam, A. I., Al-Omari, F., Roy, B., Roy, C. K., & Schneider, K. A. (2023). *LLMs for generating semantic and cross-language clones.* In IWSC.

Critique this poster

ACTIVITY

- Name one thing that works
- Name one thing you would fix
- Think message, layout, design, and presenting
- Two minutes, then we compare

CC BY NC SA

From Code Translation to Clone Detection

Palash R. Roy

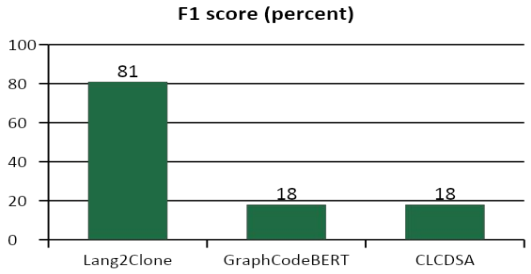
Intro

Code clones are similar fragments made by reuse
Cross-language clones are hard to find
Tools work within one language, not across

Methods

1. Translate code across languages with an LLM
2. Detect clones with a syntactic tool, NiCad
3. Map matches back to the original code
4. Test on LeetCode and GPTCloneBench

Results



96 percent precision, 71 percent recall
Best F1, well above GraphCodeBERT and CLCDSA

Discussion

Works best for structurally similar languages
Translating into C is the weakest

Translate code into a common language first, then **detect the clones.**

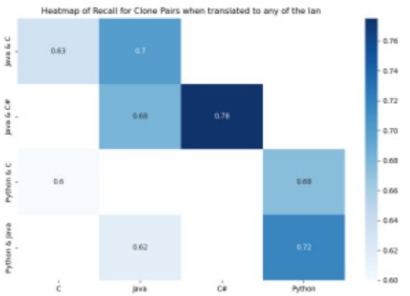
Lang2Clone reaches **81 percent F1**, far above the alternatives.

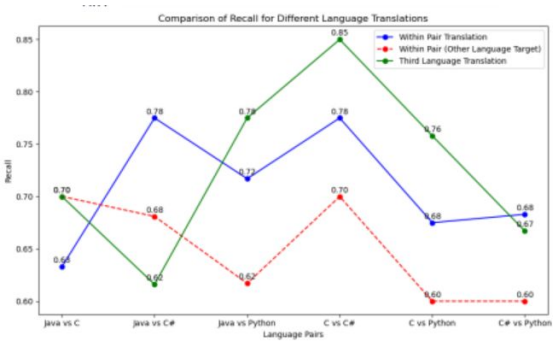
Scan for the full paper, tables, and code.

Extra Tables and Figures

Tool	Prec	Rec	F1
Lang2Clone	96	71	81
GraphCodeBERT	10	75	18
CLCDSA	84	10	18

Table. Precision, recall, F1 (percent).







Pick one thing to fix on your poster

- One message, put it in the title
- One clear path through the layout
- Big text and strong figures
- A short pitch, and open questions

CC BY NC SA

References and further reading

- Morrison, M. How to create a better research poster in less time.
- Purrington, C. Designing conference posters.
- Faulkes, Z. Better Posters blog.
- USask University Library, graduate learning supports.

CC BY NC SA

Questions/Comments

Thank you!

Grad Learning Supports

Transition Workshops

CC BY NC SA

Your university library is here to help

Writing | Math and stats | Study skills | Research

First year | Grad students | EAL | Digital skills

library.usasask.ca/studentlearning

CC BY NC SA

UNIVERSITY OF SASKATCHEWAN
University Library
LIBRARY.USASK.CA