INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

FLLM 2025
The 3rd International Conference on Foundation and Large Language Models

PARASKEVI KIVROGLOU, TIM SCHLIPPE & SIMON MARTIN

INVESTIGATING RETRIEVAL AUGMENTED GENERATION FOR LLM-BASED CODE GENERATION

Vienna, Austria
November 25, 2025

CONTENT

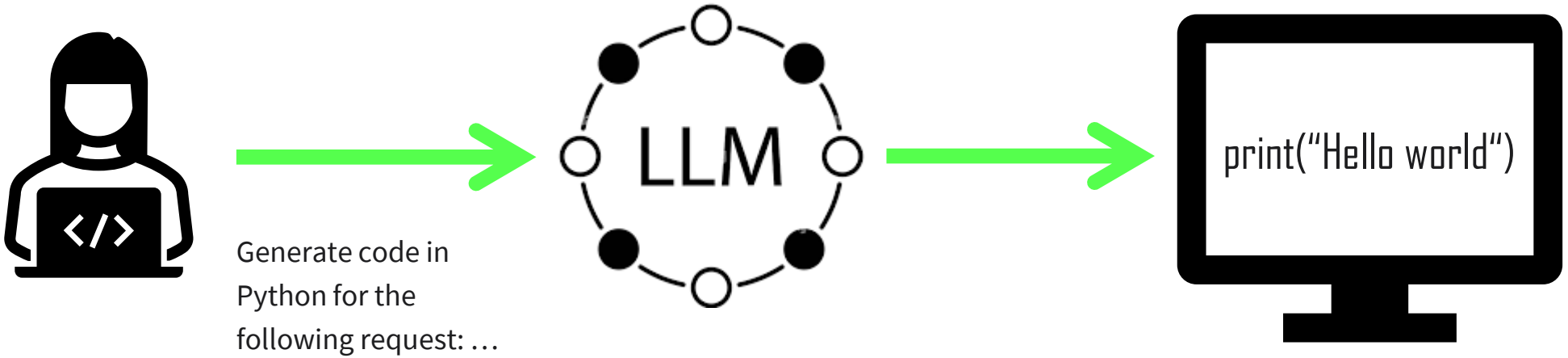
INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

1

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

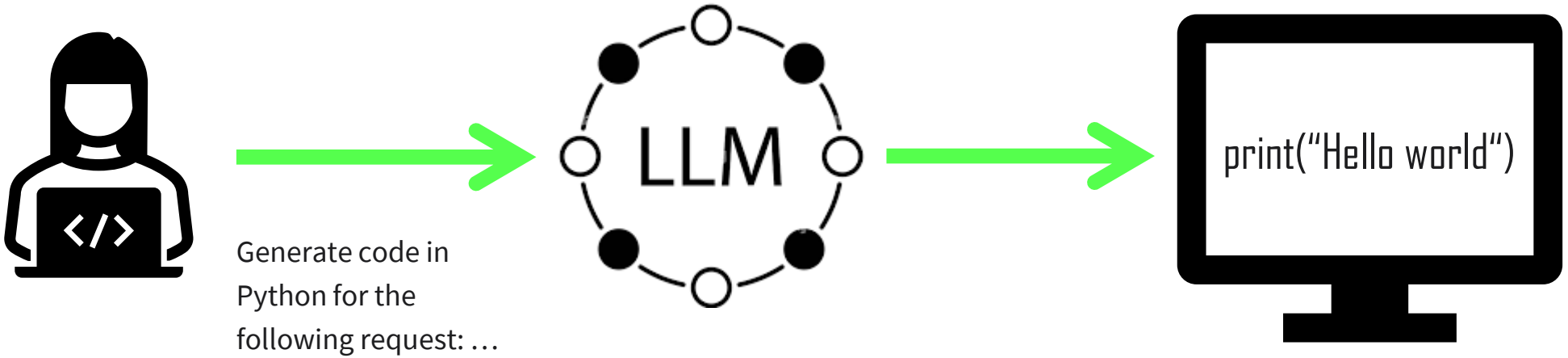
INTRODUCTION

MOTIVATION: LLMS FOR CODE GENERATION



Generate code in Python for the following request: …

MOTIVATION: LLMS FOR CODE GENERATION

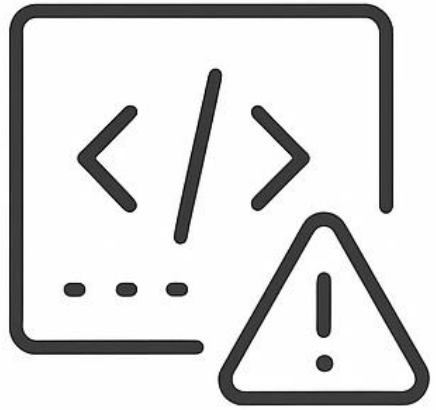


TRANSFORM NATURAL LANGUAGE INTO CODE, HIGH PERFORMANCE ON BENCHMARKS

MOTIVATION: CHALLENGES FOR LLMS

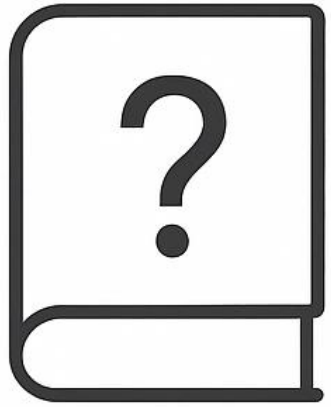
DOMAIN SPECIFIC KNOWLEDGE

UP-TO-DATE INFORMATION



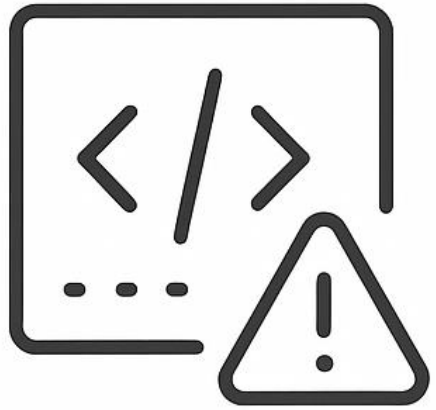
COMPLEX PROGRAMMING CONSTRUCTS

MOTIVATION: CHALLENGES FOR LLMS



DOMAIN SPECIFIC KNOWLEDGE

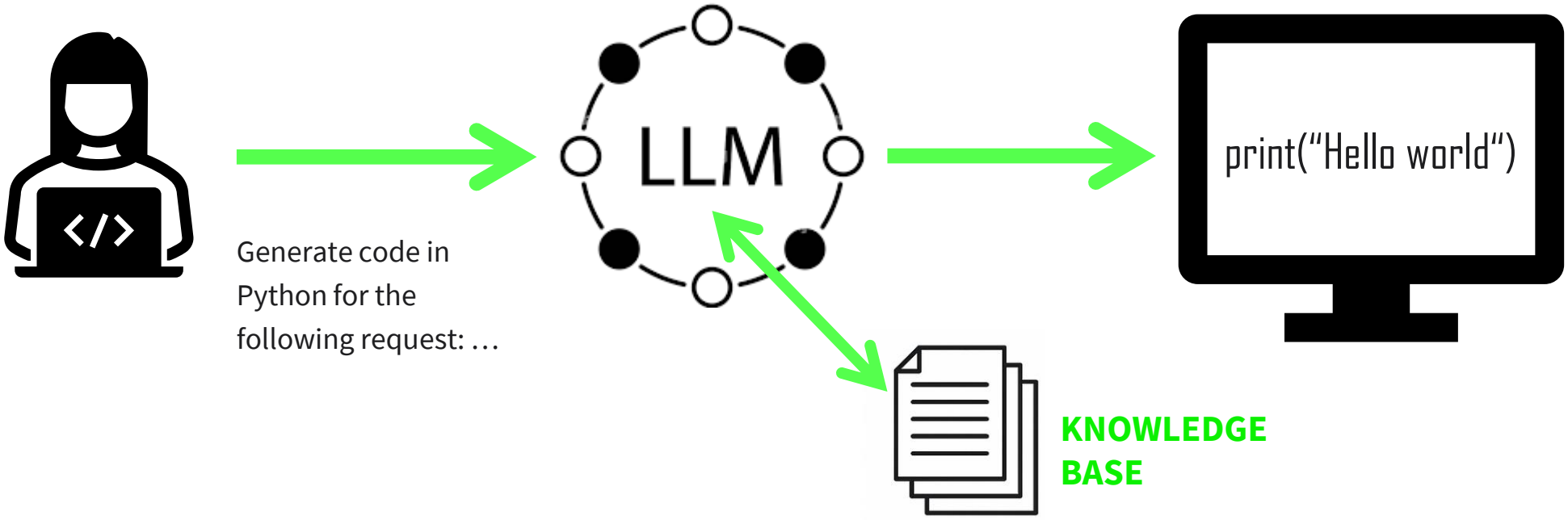
UP-TO-DATE INFORMATION



COMPLEX PROGRAMMING CONSTRUCTS

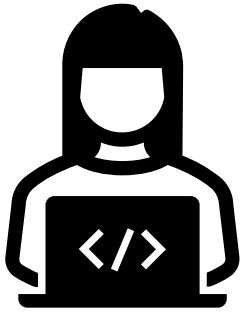
LLMS STRUGGLE WITHOUT EXTERNAL, DOMAIN-RELEVANT CONTEXT

MOTIVATION: LLMS WITH RAG



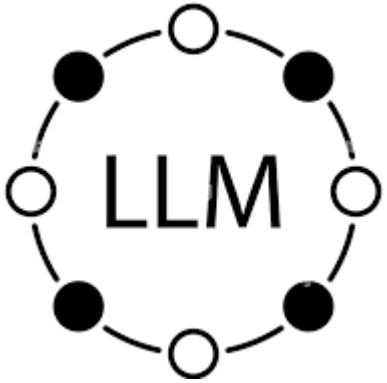
RETRIEVAL AUGMENTED GENERATION (RAG) AS PROMISING TECHNIQUE TO ENHANCE PERFORMANCE

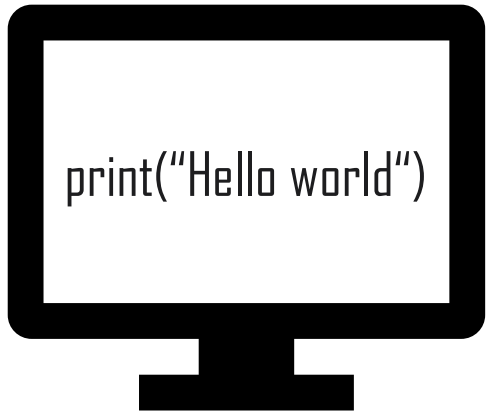
MOTIVATION: LLMS WITH RAG: SCENARIO



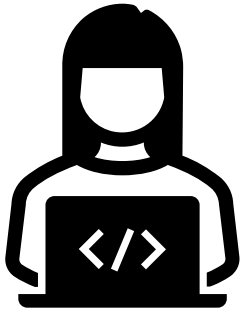
Generate code in Python for the following request: …

USER QUERY





MOTIVATION: LLMS WITH RAG: SCENARIO

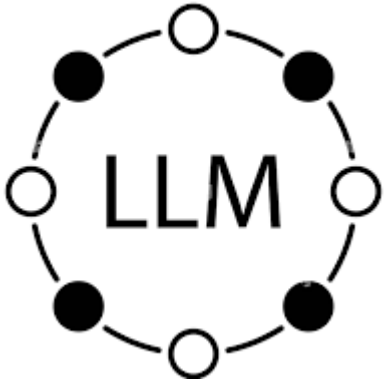


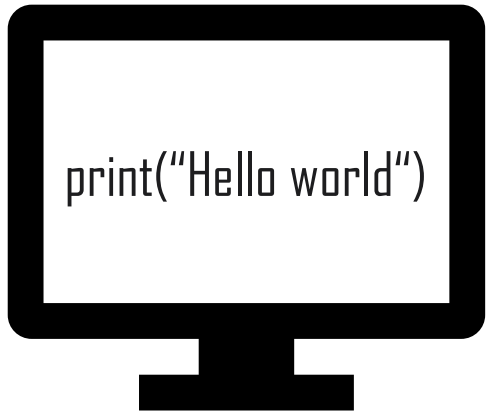
Generate code in Python for the following request: …

USER QUERY

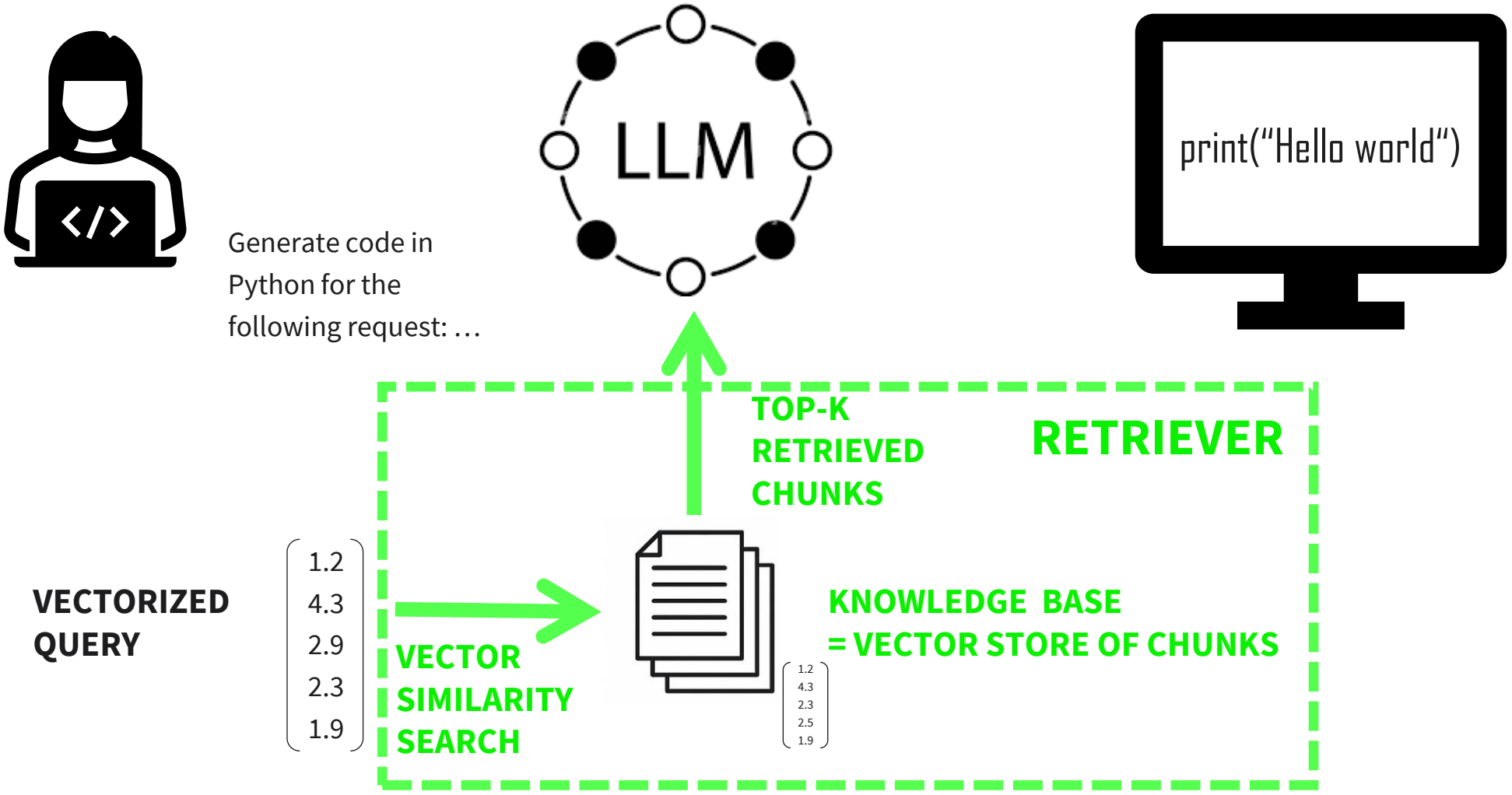
VECTORIZED QUERY

$$\begin{pmatrix} 1.2 \\ 4.3 \\ 2.9 \\ 2.3 \\ 1.9 \end{pmatrix}$$

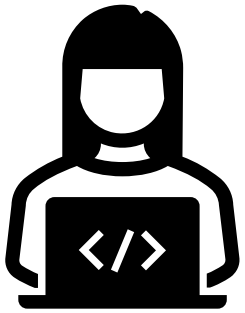




MOTIVATION: LLMS WITH RAG: SCENARIO

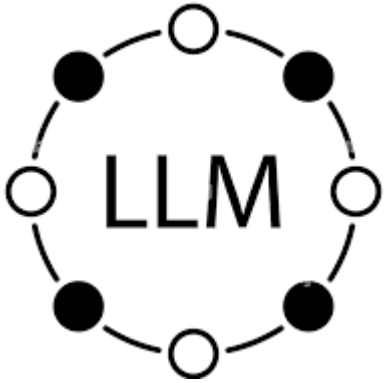


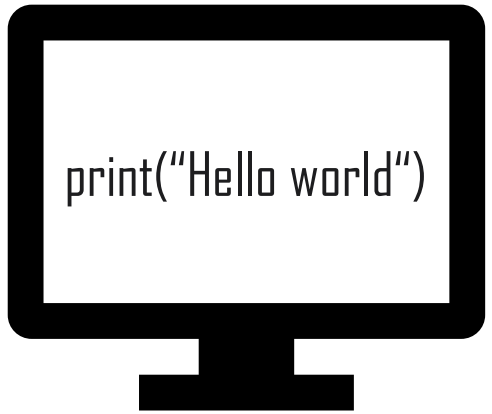
MOTIVATION: LLMS WITH RAG: SCENARIO



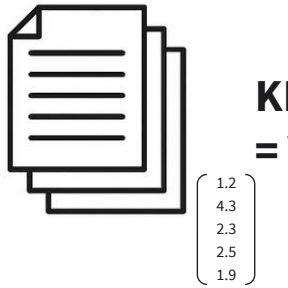
Generate code in Python for the following request: …

USER QUERY



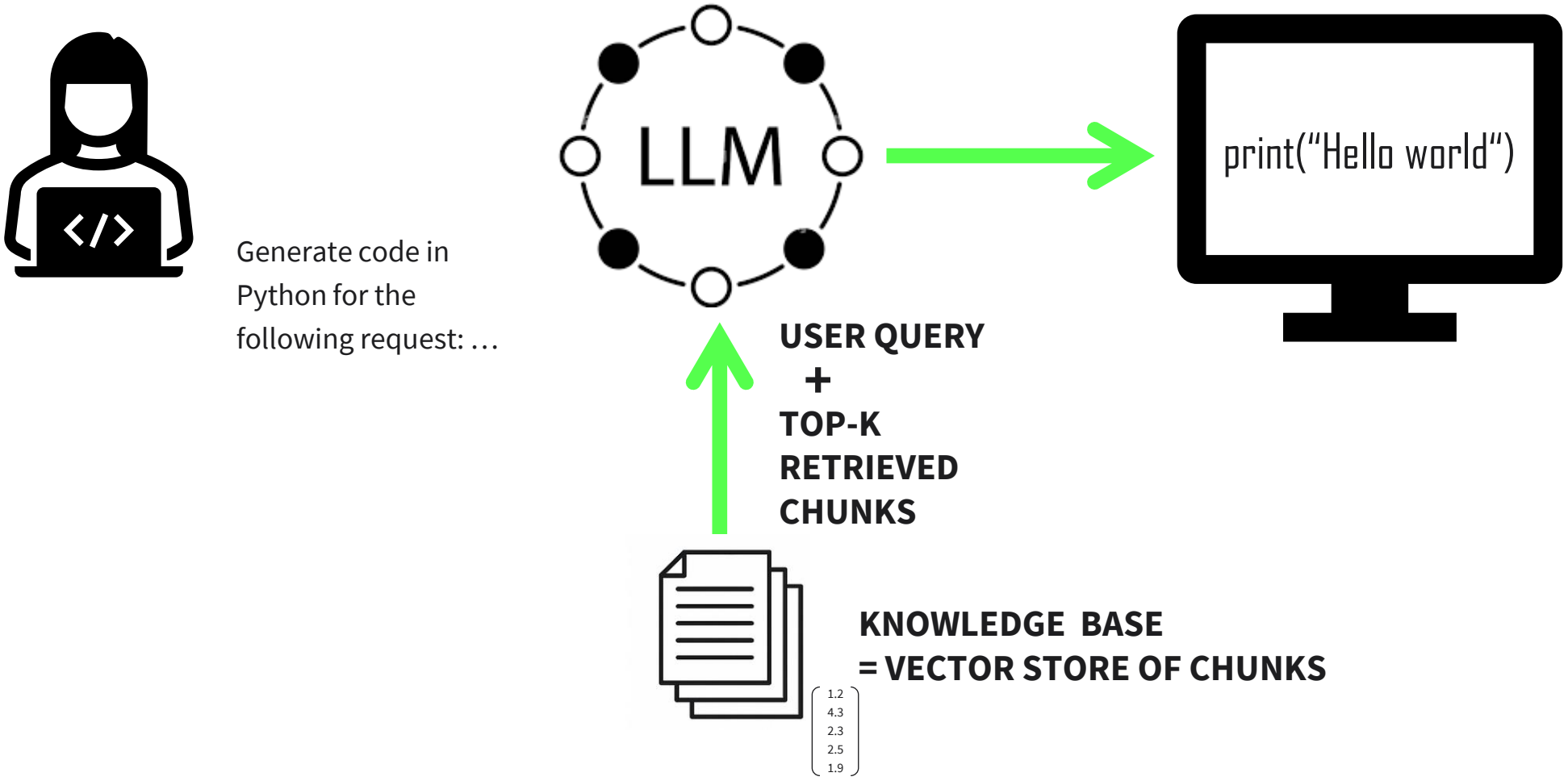


USER QUERY
+
TOP-K
RETRIEVED
CHUNKS



KNOWLEDGE BASE
= VECTOR STORE OF CHUNKS

MOTIVATION: LLMS WITH RAG: SCENARIO



2

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

RELATED WORK

RELATED WORK: LLMS IN CODE GENERATION

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

- Performance varies widely across LLMs
  (Idrisov & Schlippe, 2024)

RELATED WORK: LLMS IN CODE GENERATION

- Performance varies widely across LLMs
  (Idrisov & Schlippe, 2024)

- LLMs like Qwen2.5-Coder, CodeLlama, and GitHub Copilot generate reliable code
  (Touvron et al., 2023 ; Hui et al., 2024)

RELATED WORK: LLMS IN CODE GENERATION

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

- Performance varies widely across LLMs
  (Idrisov & Schlippe, 2024)

- LLMs like Qwen2.5-Coder, CodeLlama, and GitHub Copilot generate reliable code
  (Touvron et al., 2023 ; Hui et al., 2024)

LLMS STILL STRUGGLE WITH COMPLEX LOGIC AND DOMAIN-SPECIFIC KNOWLEDGE *(Huynh & Lin, 2025)*

RELATED WORK: RAG-BASED CODE GENERATION

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

- External knowledge boosts code generation accuracy
  (Wang et al., 2024)

RELATED WORK: RAG-BASED CODE GENERATION

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

- External knowledge boosts code generation accuracy
  (Wang et al., 2024)
- No single RAG configuration (LLM & retrieval methods) works best for all tasks or cost constraints
  (Yang et al., 2025)

RELATED WORK: RAG-BASED CODE GENERATION

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

- External knowledge boosts code generation accuracy
  (Wang et al., 2024)
- No single RAG configuration (LLM & retrieval methods) works best for all tasks or cost constraints
  (Yang et al., 2025)
- Some RAG systems update and refine their results across multiple steps.
  (Su et al., 2024; Ma et al., 2025)

RELATED WORK: RAG-BASED CODE GENERATION

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

- External knowledge boosts code generation accuracy
  (Wang et al., 2024)
- No single RAG configuration (LLM & retrieval methods) works best for all tasks or cost constraints
  (Yang et al., 2025)
- Some RAG systems update and refine their results across multiple steps.
  (Su et al., 2024; Ma et al., 2025)

EFFECT OF RETRIEVAL CORPUS SIZE ON CODE GENERATION IS UNEXPLORED

RELATED WORK: SELF-EVALUATION OF LLMS

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

- Self-evaluation methods allow LLMs iteratively critique and revise responses
  (Madaan et al., 2023)

RELATED WORK: SELF-EVALUATION OF LLMS

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

- Self-evaluation methods allow LLMs iteratively critique and revise responses
  (Madaan et al., 2023)

- Collaborative multi-agent systems and specialized prompting techniques improve quality of generated code
  (Quoc et al., 2024; Zhang et al., 2024)

RELATED WORK: SELF-EVALUATION OF LLMS

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

- Self-evaluation methods allow LLMs iteratively critique and revise responses
  (Madaan et al., 2023)

- Collaborative multi-agent systems and specialized prompting techniques improve quality of generated code
  (Quoc et al., 2024; Zhang et al., 2024)

COMBINATION OF SELF-EVALUATION WITH RAG REMAINS UNEXPLORED FOR LLM-BASED CODE GENERATION

3

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

EXPERIMENTAL SETUP

EXPERIMENTAL SETUP: RAG SCENARIOS

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

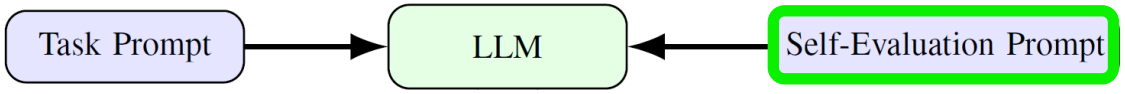
BASELINE

Scenario	Key Prompting Elements
1a	Base prompt: "Generate code in Python for the following request: [coding problem]"

EXPERIMENTAL SETUP: RAG SCENARIOS

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

SELF-EVALUATION

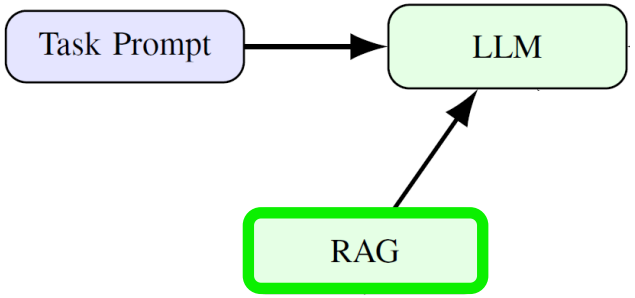


Scenario	Key Prompting Elements
1a	Base prompt: "Generate code in Python for the following request: [coding problem]"
1b	1a + Self-evaluation: "After generating code, evaluate for: (1) Logical errors (2) Edge cases (3) Syntax errors (4) Performance issues"

EXPERIMENTAL SETUP: RAG SCENARIOS

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

BASIC RAG

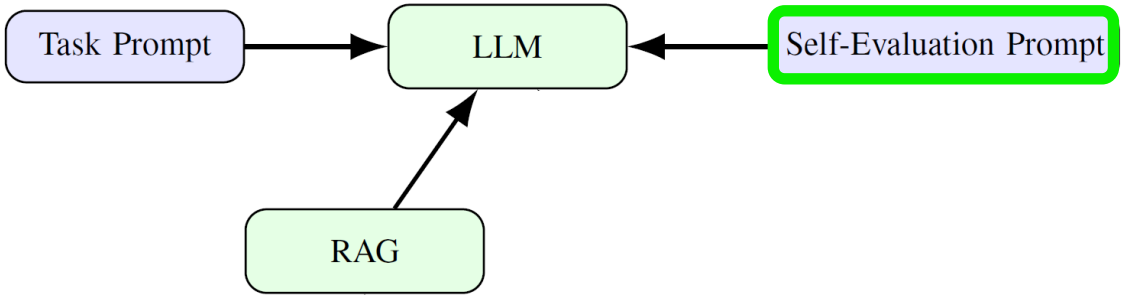


Scenario	Key Prompting Elements
1a	Base prompt: "Generate code in Python for the following request: [coding problem]"
1b	1a + Self-evaluation: "After generating code, evaluate for: (1) Logical errors (2) Edge cases (3) Syntax errors (4) Performance issues"
2a	1a + Retrieval: "Use the following retrieved code examples to answer the coding question: [retrieved code snippets]"

EXPERIMENTAL SETUP: RAG SCENARIOS

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

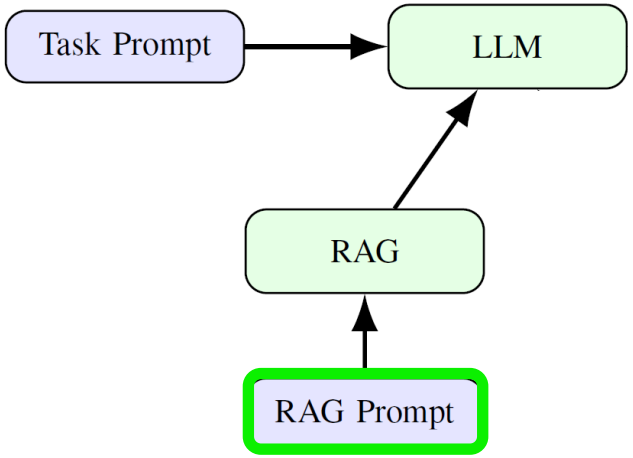
RAG WITH SELF-EVALUATION



Scenario	Key Prompting Elements
1a	Base prompt: "Generate code in Python for the following request: [coding problem]"
1b	1a + Self-evaluation: "After generating code, evaluate for: (1) Logical errors (2) Edge cases (3) Syntax errors (4) Performance issues"
2a	1a + Retrieval: "Use the following retrieved code examples to answer the coding question: [retrieved code snippets]"
2b	2a + Self-evaluation: "After generating code using retrieved examples, evaluate and improve your solution for correctness, efficiency, and readability"

EXPERIMENTAL SETUP: RAG SCENARIOS

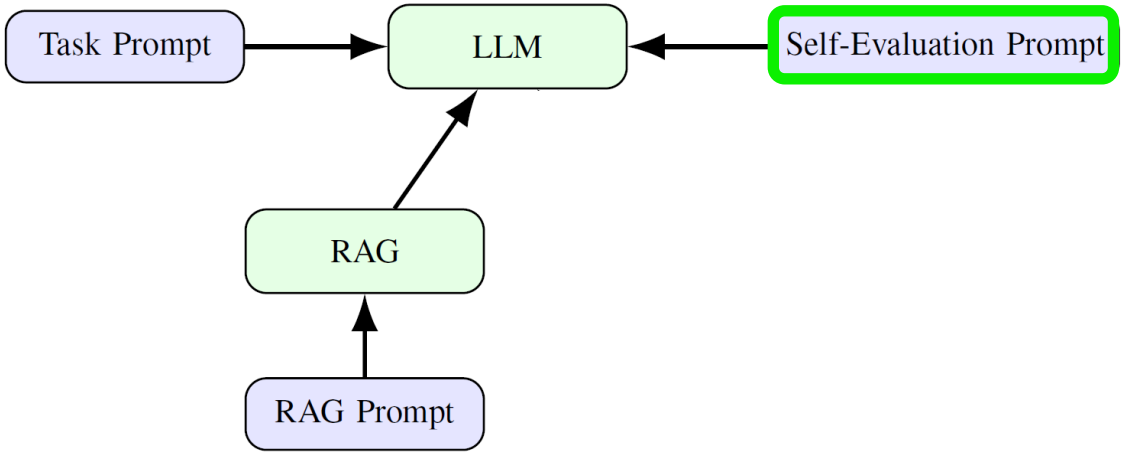
RAG WITH SPECIFIC PROMPTS



Scenario	Key Prompting Elements
1a	Base prompt: "Generate code in Python for the following request: [coding problem]"
1b	1a + Self-evaluation: "After generating code, evaluate for: (1) Logical errors (2) Edge cases (3) Syntax errors (4) Performance issues"
2a	1a + Retrieval: "Use the following retrieved code examples to answer the coding question: [retrieved code snippets]"
2b	2a + Self-evaluation: "After generating code using retrieved examples, evaluate and improve your solution for correctness, efficiency, and readability"
3a	2a + Specific prompts: "Use retrieval tool to find relevant code. Follow structured Thought/Action/Observation format"

EXPERIMENTAL SETUP: RAG SCENARIOS

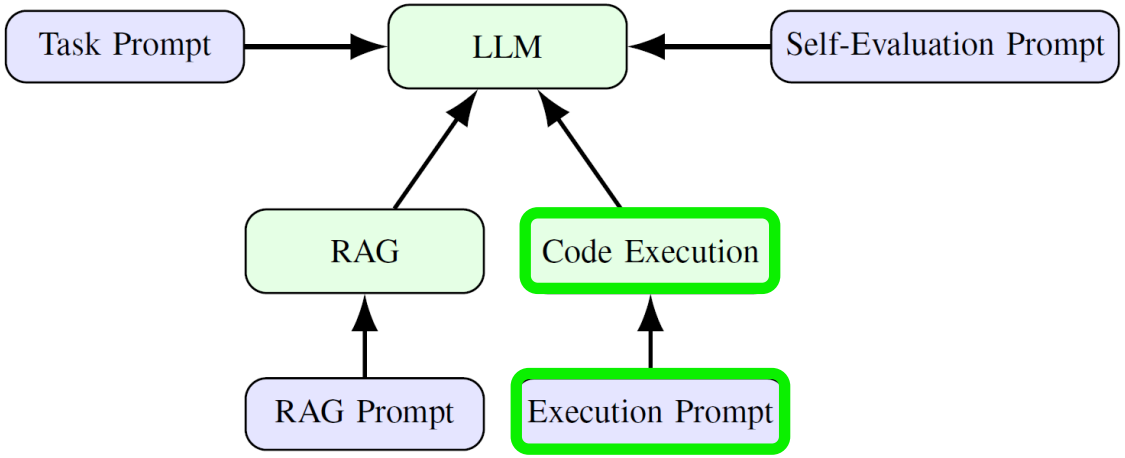
ENHANCED RAG WITH SELF-EVALUATION



Scenario	Key Prompting Elements
1a	Base prompt: "Generate code in Python for the following request: [coding problem]"
1b	1a + Self-evaluation: "After generating code, evaluate for: (1) Logical errors (2) Edge cases (3) Syntax errors (4) Performance issues"
2a	1a + Retrieval: "Use the following retrieved code examples to answer the coding question: [retrieved code snippets]"
2b	2a + Self-evaluation: "After generating code using retrieved examples, evaluate and improve your solution for correctness, efficiency, and readability"
3a	2a + Specific prompts: "Use retrieval tool to find relevant code. Follow structured Thought/Action/Observation format"
3b	3a + Self-evaluation: "After generating code, verify and correct for logical errors, edge cases, syntax errors, and performance issues"

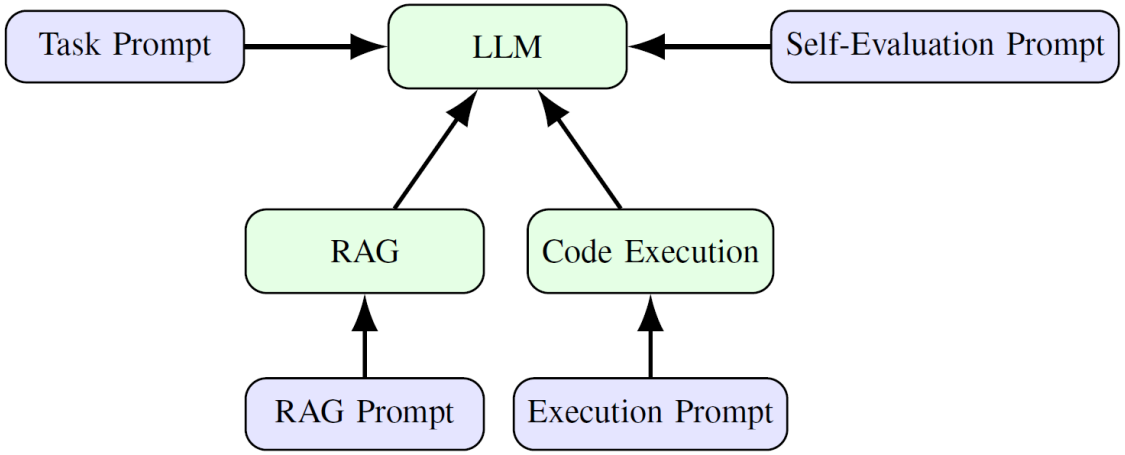
EXPERIMENTAL SETUP: RAG SCENARIOS

ENHANCED RAG WITH SELF-EVALUATION AND CODE EXECUTION



Scenario	Key Prompting Elements
1a	Base prompt: "Generate code in Python for the following request: [coding problem]"
1b	1a + Self-evaluation: "After generating code, evaluate for: (1) Logical errors (2) Edge cases (3) Syntax errors (4) Performance issues"
2a	1a + Retrieval: "Use the following retrieved code examples to answer the coding question: [retrieved code snippets]"
2b	2a + Self-evaluation: "After generating code using retrieved examples, evaluate and improve your solution for correctness, efficiency, and readability"
3a	2a + Specific prompts: "Use retrieval tool to find relevant code. Follow structured Thought/Action/Observation format"
3b	3a + Self-evaluation: "After generating code, verify and correct for logical errors, edge cases, syntax errors, and performance issues"
4	3b + Code execution: "Execute code against test cases and iteratively improve based on results (max 3 cycles)"

EXPERIMENTAL SETUP: RAG SCENARIOS

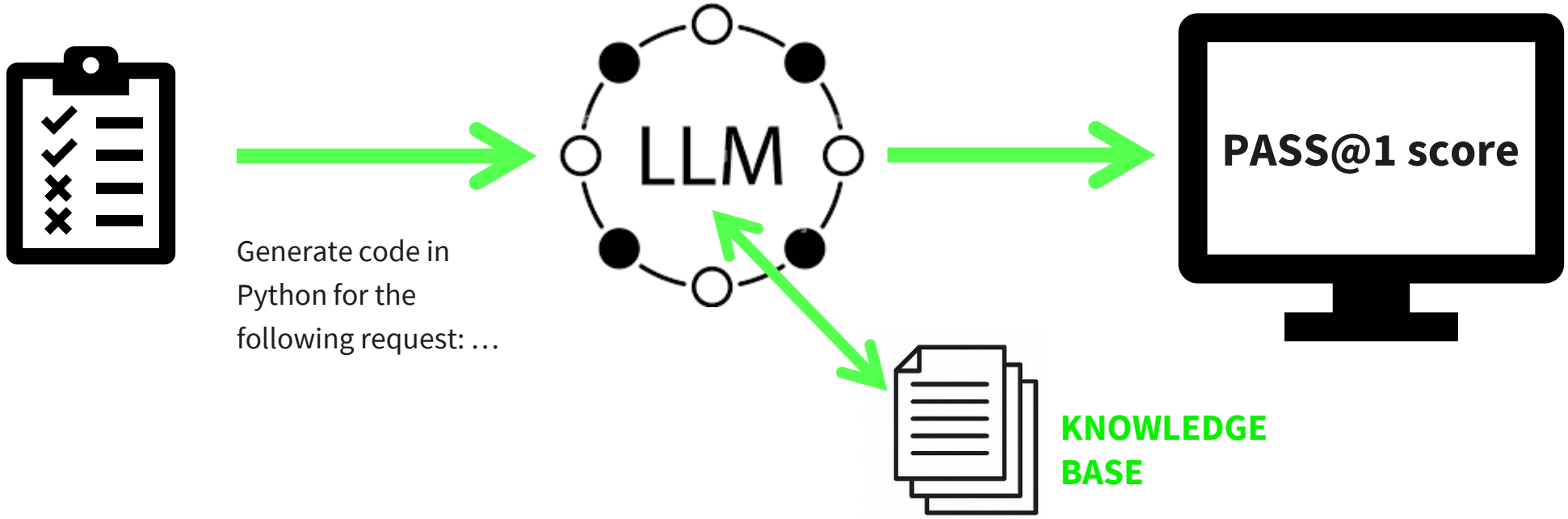


LANGCHAIN

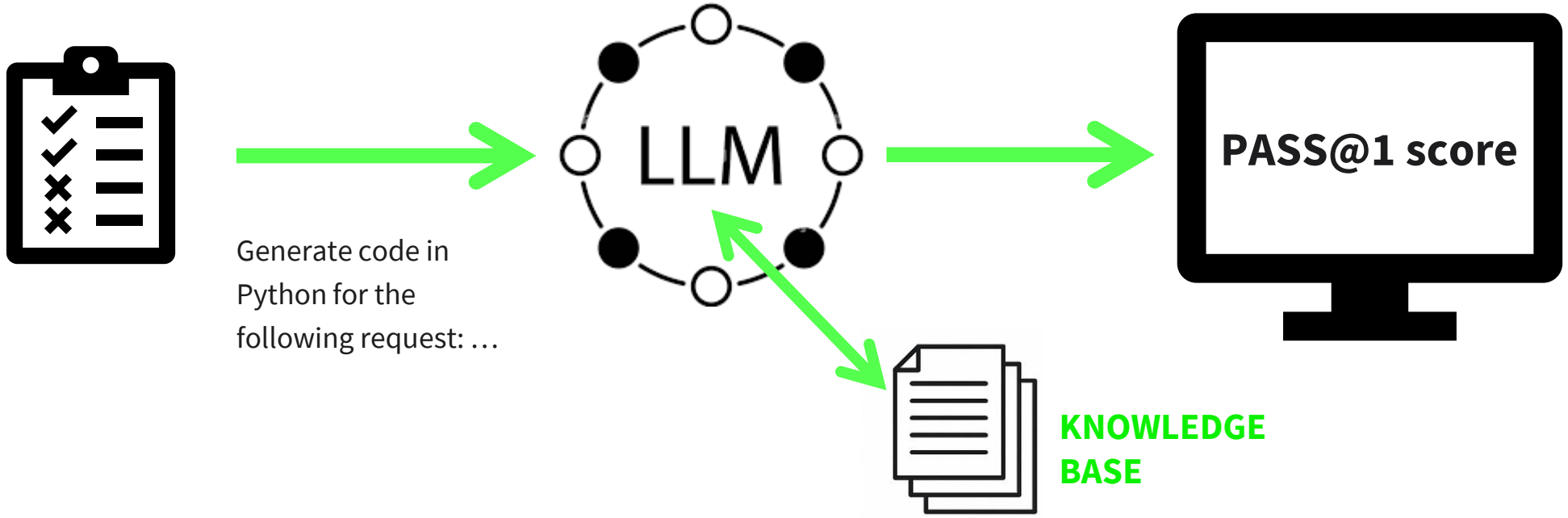
https://github.com/langchain-ai/langchain

Scenario	Key Prompting Elements
1a	Base prompt: "Generate code in Python for the following request: [coding problem]"
1b	1a + Self-evaluation: "After generating code, evaluate for: (1) Logical errors (2) Edge cases (3) Syntax errors (4) Performance issues"
2a	1a + Retrieval: "Use the following retrieved code examples to answer the coding question: [retrieved code snippets]"
2b	2a + Self-evaluation: "After generating code using retrieved examples, evaluate and improve your solution for correctness, efficiency, and readability"
3a	2a + Specific prompts: "Use retrieval tool to find relevant code. Follow structured Thought/Action/Observation format"
3b	3a + Self-evaluation: "After generating code, verify and correct for logical errors, edge cases, syntax errors, and performance issues"
4	3b + Code execution: "Execute code against test cases and iteratively improve based on results (max 3 cycles)"

EXPERIMENTAL SETUP: LLM, DATA, EVALUATION



EXPERIMENTAL SETUP: LLM, DATA, EVALUATION

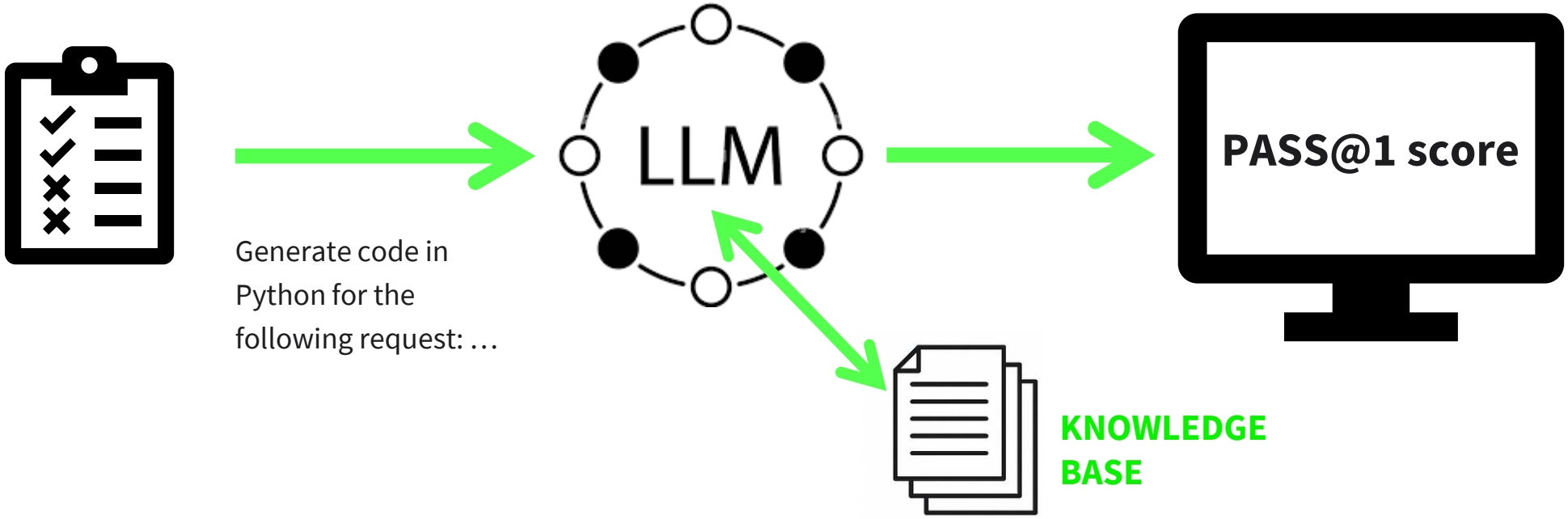


MBPP+

(Liu et al., 2023; 2024)

- Evaluates functional correctness & edge cases
- Suitable for Python code generation

EXPERIMENTAL SETUP: LLM, DATA, EVALUATION



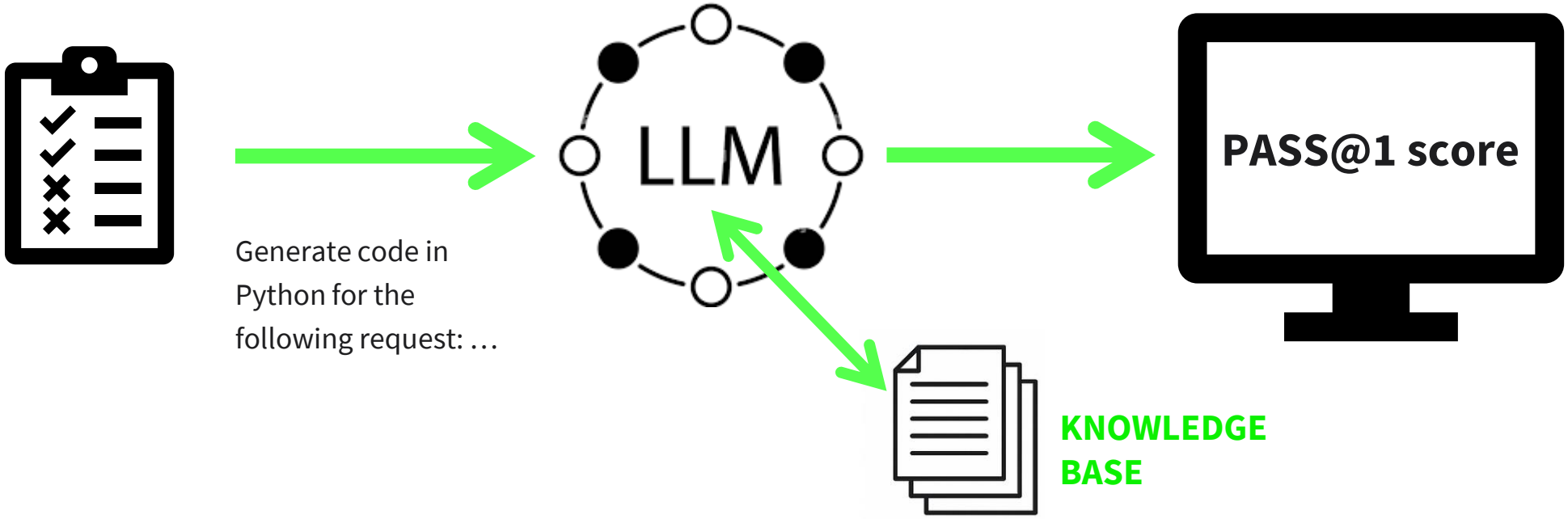
MBPP+

Qwen2.5-Coder-32B-Instruct

(Huynh & Lin, 2025)

- State-of-the-art code generation model
- Strong performance

EXPERIMENTAL SETUP: LLM, DATA, EVALUATION



MBPP+

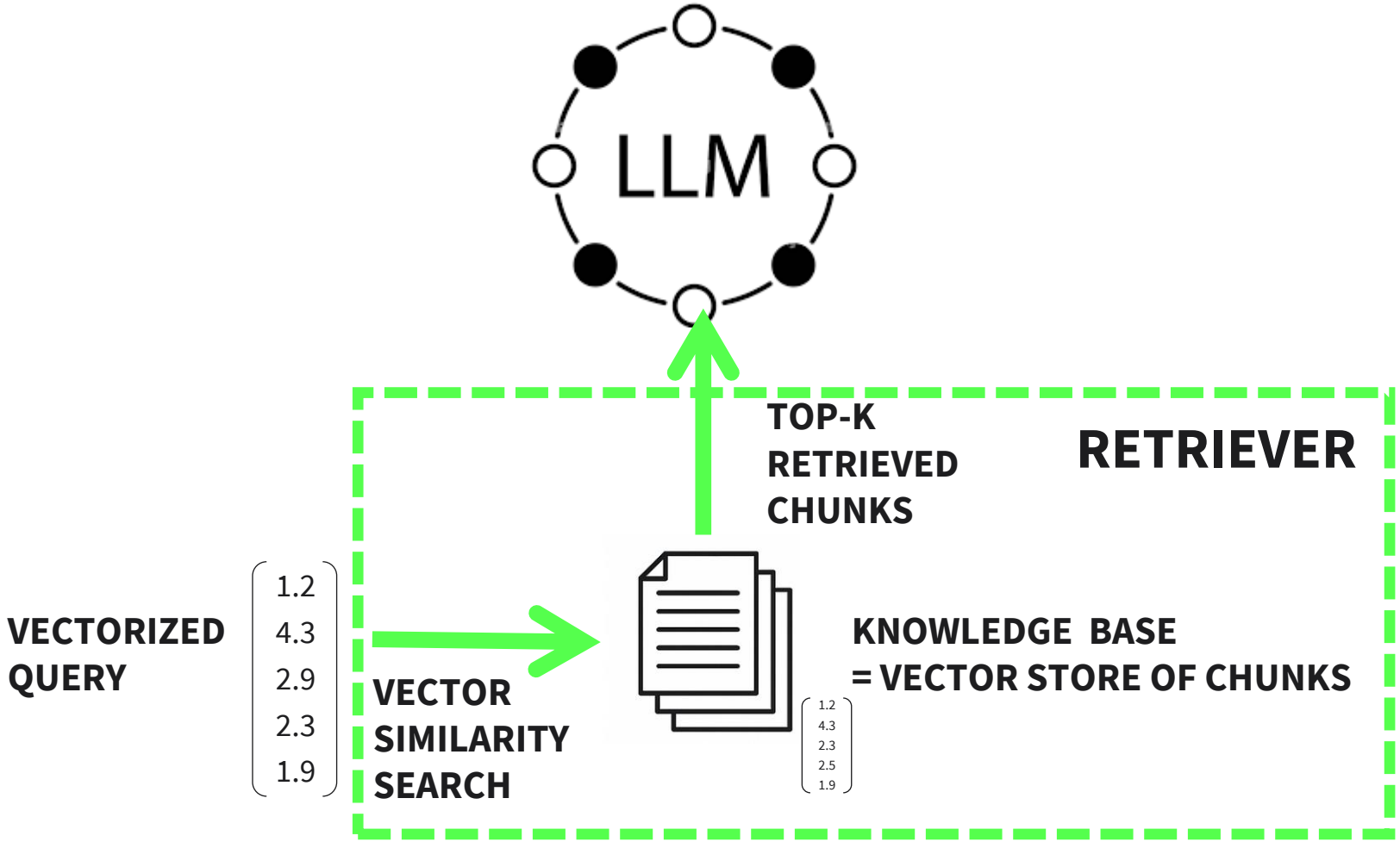
Qwen2.5-Coder-32B-Instruct

CodeSearchNet

(Husain et al., 2020; Günther et al., 2024)

- Enables retrieval of relevant code examples
- Python subset of 1k / 2k / 3k functions
- Embedded using jina-embeddings-v2-base-en

EXPERIMENTAL SETUP: RAG IMPLEMENTATION



- Fetches **10** candidates, keeps the top **k=8**
- Balances relevance and diversity using **λ = 0.85**
- Maximal Marginal Relevance (**MMR**) reranking ensures the LLM gets useful, non-redundant examples

4

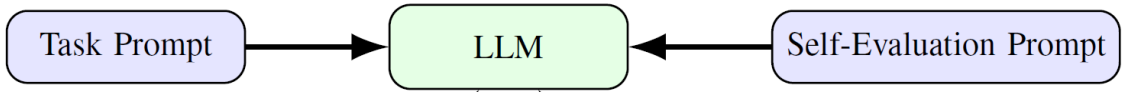
INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

RESULTS

EXPERIMENTAL SETUP: RAG SCENARIOS

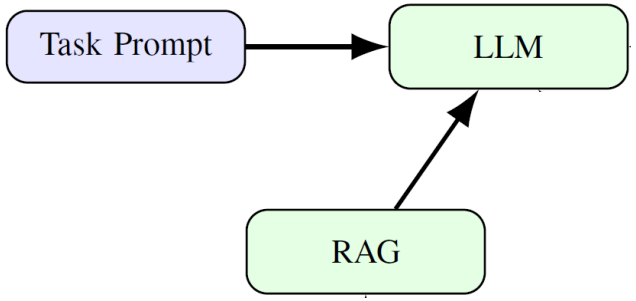
Scenario		1k documents	2k documents	3k documents
1a	Baseline	83.60%	83.60%	83.60%

EXPERIMENTAL SETUP: RAG SCENARIOS



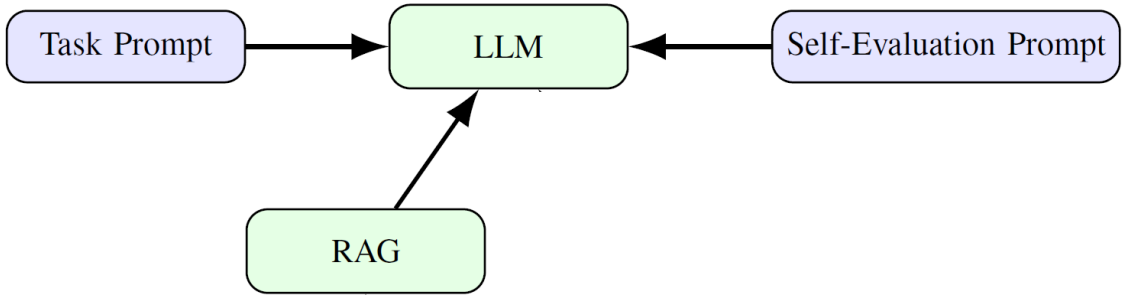
Scenario		1k documents	2k documents	3k documents
1a	Baseline	83.60%	83.60%	83.60%
1b	Self-Evaluation	77.25%	77.25%	77.25%

EXPERIMENTAL SETUP: RAG SCENARIOS



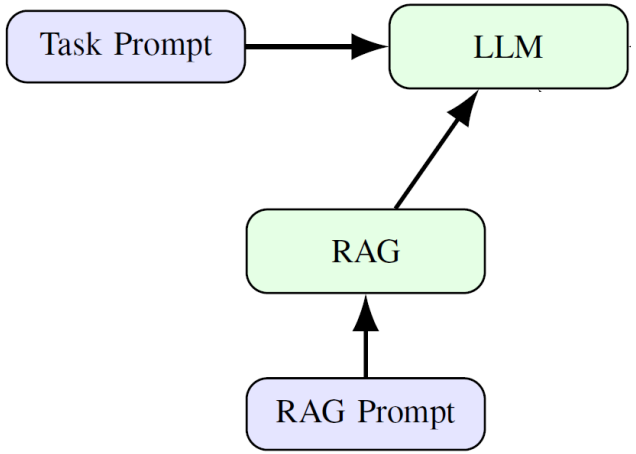
Scenario		1k documents	2k documents	3k documents
1a	Baseline	83.60%	83.60%	83.60%
1b	Self-Evaluation	77.25%	77.25%	77.25%
2a	Basic RAG	82.54%	84.13%	86.78%

EXPERIMENTAL SETUP: RAG SCENARIOS



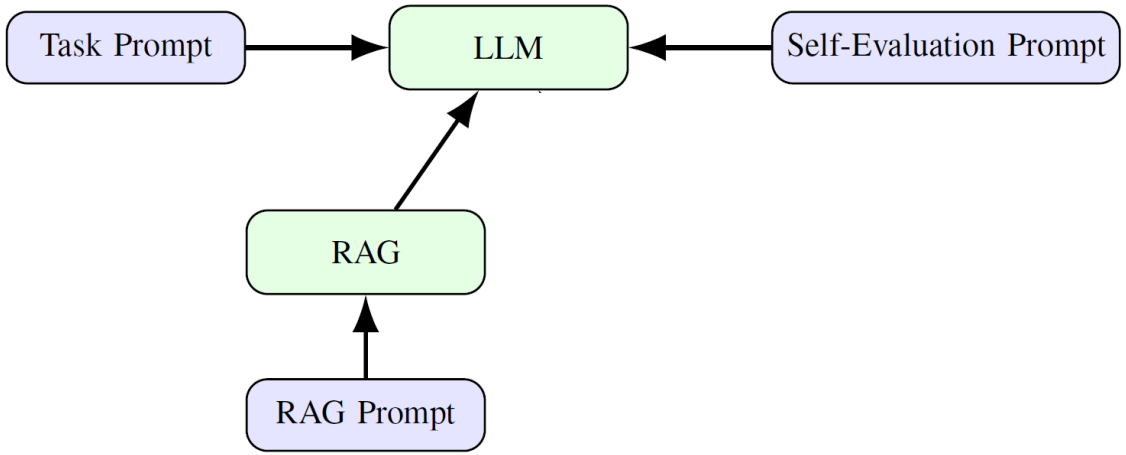
Scenario		1k documents	2k documents	3k documents
1a	Baseline	83.60%	83.60%	83.60%
1b	Self-Evaluation	77.25%	77.25%	77.25%
2a	Basic RAG	82.54%	84.13%	86.78%
2b	RAG with Self-Evaluation	84.92%	85.72%	89.00%

EXPERIMENTAL SETUP: RAG SCENARIOS



Scenario		1k documents	2k documents	3k documents
1a	Baseline	83.60%	83.60%	83.60%
1b	Self-Evaluation	77.25%	77.25%	77.25%
2a	Basic RAG	82.54%	84.13%	86.78%
2b	RAG with Self-Evaluation	84.92%	85.72%	89.00%
3a	RAG with Specific Prompts	88.62%	87.30%	87.84%

EXPERIMENTAL SETUP: RAG SCENARIOS



Scenario		1k documents	2k documents	3k documents
1a	Baseline	83.60%	83.60%	83.60%
1b	Self-Evaluation	77.25%	77.25%	77.25%
2a	Basic RAG	82.54%	84.13%	86.78%
2b	RAG with Self-Evaluation	84.92%	85.72%	89.00%
3a	RAG with Specific Prompts	88.62%	87.30%	87.84%
3b	Enhanced RAG with Self-Evaluation	90.74%	91.54%	**92.06%**

6

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

CONCLUSION AND FUTURE WORK

CONCLUSION AND FUTURE WORK

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

Conclusion

- **RAG significantly improves code generation accuracy**
  - ➔ Up to +10.11% pass@1 compared to the baseline

CONCLUSION AND FUTURE WORK

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

Conclusion

- **RAG significantly improves code generation accuracy**
  - ➔ Up to +10.11% pass@1 compared to the baseline
- **Self-evaluation alone can reduce performance**
  - ➔ But in combination with RAG, it leads to substantial gains

CONCLUSION AND FUTURE WORK

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

Conclusion

- **RAG significantly improves code generation accuracy**
  - ➔ Up to +10.11% pass@1 compared to the baseline
- **Self-evaluation alone can reduce performance**
  - ➔ But in combination with RAG, it leads to substantial gains
- **Specific RAG prompting strategies are essential**
  - ➔ They improve the LLM's ability to use retrieved examples effectively

CONCLUSION AND FUTURE WORK

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

Conclusion

- **RAG significantly improves code generation accuracy**
  - ➔ Up to +10.11% pass@1 compared to the baseline
- **Self-evaluation alone can reduce performance**
  - ➔ But in combination with RAG, it leads to substantial gains
- **Specific RAG prompting strategies are essential**
  - ➔ They improve the LLM's ability to use retrieved examples effectively
- **Retrieval corpus size matters**
  - ➔ Larger code corpora (3k functions) yield consistently better results

CONCLUSION AND FUTURE WORK

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

Conclusion

- **RAG significantly improves code generation accuracy**
  - ➔ Up to +10.11% pass@1 compared to the baseline
- **Self-evaluation alone can reduce performance**
  - ➔ But in combination with RAG, it leads to substantial gains
- **Specific RAG prompting strategies are essential**
  - ➔ They improve the LLM's ability to use retrieved examples effectively
- **Retrieval corpus size matters**
  - ➔ Larger code corpora (3k functions) yield consistently better results
- **Execution feedback adds only marginal benefits**
  - ➔ Most gains are already captured by RAG + self-evaluation + structured prompts

CONCLUSION AND FUTURE WORK

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

Conclusion

- **RAG significantly improves code generation accuracy**
  - ➔ Up to +10.11% pass@1 compared to the baseline
- **Self-evaluation alone can reduce performance**
  - ➔ But in combination with RAG, it leads to substantial gains
- **Specific RAG prompting strategies are essential**
  - ➔ They improve the LLM's ability to use retrieved examples effectively
- **Retrieval corpus size matters**
  - ➔ Larger code corpora (3k functions) yield consistently better results
- **Execution feedback adds only marginal benefits**
  - ➔ Most gains are already captured by RAG + self-evaluation + structured prompts

Future Work

- **Develop specialized code embedding models**
  - ➔ To improve retrieval quality for code-specific patterns

CONCLUSION AND FUTURE WORK

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

Conclusion

- **RAG significantly improves code generation accuracy**
  - ➔ Up to +10.11% pass@1 compared to the baseline
- **Self-evaluation alone can reduce performance**
  - ➔ But in combination with RAG, it leads to substantial gains
- **Specific RAG prompting strategies are essential**
  - ➔ They improve the LLM's ability to use retrieved examples effectively
- **Retrieval corpus size matters**
  - ➔ Larger code corpora (3k functions) yield consistently better results
- **Execution feedback adds only marginal benefits**
  - ➔ Most gains are already captured by RAG + self-evaluation + structured prompts

Future Work

- **Develop specialized code embedding models**
  - ➔ To improve retrieval quality for code-specific patterns
- **Adaptive retrieval strategies**
  - ➔ techniques to dynamically adjust retrieval parameters based on query characteristics

CONCLUSION AND FUTURE WORK

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

Conclusion

- **RAG significantly improves code generation accuracy**
  - ➔ Up to +10.11% pass@1 compared to the baseline
- **Self-evaluation alone can reduce performance**
  - ➔ But in combination with RAG, it leads to substantial gains
- **Specific RAG prompting strategies are essential**
  - ➔ They improve the LLM's ability to use retrieved examples effectively
- **Retrieval corpus size matters**
  - ➔ Larger code corpora (3k functions) yield consistently better results
- **Execution feedback adds only marginal benefits**
  - ➔ Most gains are already captured by RAG + self-evaluation + structured prompts

Future Work

- **Develop specialized code embedding models**
  - ➔ To improve retrieval quality for code-specific patterns
- **Adaptive retrieval strategies**
  - ➔ techniques to dynamically adjust retrieval parameters based on query characteristics
- **Improved self-evaluation techniques for code**
  - ➔ using static analysis or symbolic execution to improve code quality

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

THANK YOU

Tim Schlippe
tim.schlippe@iu.org

REFERENCES

[1] N. Huynh and B. Lin, "A Survey On Large Language Models For Code Generation," *arXiv:2503.01245*, 2025.

[2] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel *et al.*, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 9459–9474.

[3] B. Hui, J. Yang, Z. Cui, J. Yang, D. Liu, L. Zhang, T. Liu, J. Zhang, B. Yu, K. Lu *et al.*, "Qwen2.5-Coder Technical Report," *arXiv:2409.12186*, 2024.

[4] X. Li, Y. Gong, Y. Shen, X. Qiu, H. Zhang, B. Yao, W. Qi, D. Jiang, W. Chen, and N. Duan, "CodeRetriever: A Large Scale Contrastive Pre-Training Method for Code Search," in *EMNLP*, 2022, pp. 2898–2910.

[5] B. Idrisov and T. Schlippe, "Program Code Generation with Generative AIs," *Algorithms*, vol. 17, no. 2, p. 62, 2024.

[6] A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang *et al.*, "Self-Refine: Iterative Refinement with Self-Feedback," *arXiv:2303.17651*, 2023.

[7] B. Idrisov, E. Eisenacher, and T. Schlippe, "Program Code Generation: Single LLMs vs. Multi-Agent Systems," in *ICNLP*, Guangzhou, China, 2025.

[8] T. T. Quoc, D. H. Minh, T. Q. Thanh, and A. Nguyen-Duc, "An Empirical Study on Self-correcting Large Language Models for Data Science Code Generation," *arXiv:2408.15658*, 2024.

[9] W. Zhang, Y. Shen, L. Wu, Q. Peng, J. Wang, Y. Zhuang, and W. Lu, "Self-Contrast: Better Reflection Through Inconsistent Solving Perspectives," in *ACL*, 2024, pp. 3602–3622.

[10] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, and H. Wang, "Retrieval-Augmented Generation for Large Language Models: A Survey," *arXiv:2312.10997*, 2023.

[11] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, "Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection," *arXiv:2310.11511*, 2023.

[12] A. Seabra, C. Cavalcante, J. Nepomuceno, L. Lago, N. Ruberg, and S. Lifschitz, "Dynamic Multi-Agent Orchestration and Retrieval for Multi-Source Question-Answer Systems using Large Language Models," *arXiv:2412.17964*, 2024.

[13] P. Gupta, A. Ranjan, and R. R. Singh, "A Comprehensive Survey of Retrieval-Augmented Generation (RAG): Evolution, Current Landscape and Future Directions," *arXiv preprint arXiv:2410.12837*, 2024.

[14] Z. Wang, J. Qiu, J. Wang, and B. Y. Lin, "CodeRAG-Bench: Can Retrieval Augment Code Generation?" *arXiv preprint arXiv:2406.14497*, 2024.

[15] Z. Yang, Y. Hu, Q. Liu, Z. Liu, Y. He, B. Chen, and J. Zhang, "An Empirical Study of Retrieval-Augmented Code Generation: Challenges and Opportunities," *arXiv preprint arXiv:2501.13742*, 2025.

[16] H. Li, L. Zhang, M. Zhao, J. Xu, L. Zhou, and D. Zhang, "A Deep Dive into Retrieval-Augmented Generation for Code Completion: Experience on WeChat," *arXiv preprint arXiv:2507.18515*, 2025.

[17] Z. Wang, B. Sun, Z. Liu, Z. Xu, T. Zhang, and E. Chen, "RAG or Fine-tuning? A Comparative Study on LCMs-based Code Completion in Industry," *arXiv preprint arXiv:2505.15179*, 2025.

[18] Y. Su, C. Xia, J. Zhang, M. Huang, and N. Peng, "EVOR: Evolving Retrieval for Code Generation," *arXiv preprint arXiv:2402.12317*, 2024.

REFERENCES

[19] H. Ma *et al.*, "ARCS: Agentic Retrieval-Augmented Code Synthesis with Iterative Refinement," *arXiv preprint arXiv:2504.20434*, 2025.

[20] Y. Tang, X. Li, H. Pan, K. Li, and Z. Yu, "Retrieval-Augmented Code Generation for Situated Action Generation: A Case Study on Minecraft," in *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024, pp. 11 052–11 063.

[21] H. Husain, H.-H. Wu, T. Gazit, M. Allamanis, and M. Brockschmidt, "CodeSearchNet Challenge: Evaluating the State of Semantic Code Search," *arXiv:1909.09436*, 2020.

[22] J. Austin, A. Odena, M. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. Cai, M. Terry, Q. Le *et al.*, "Program Synthesis with Large Language Models," *arXiv:2108.07732*, 2021.

[23] J. Liu, C. S. Xia, Y. Wang, and L. Zhang, "Is Your Code Generated by ChatGPT Really Correct? Rigorous Evaluation of Large Language Models for Code Generation," in *NeurIPS*, 2023.

[24] J. Liu, S. Xie, J. Wang, Y. Wei, Y. Ding, and L. Zhang, "Evaluating Language Models for Efficient Code Generation," *arXiv:2408.06450*, 2024.

[25] X. Du, M. Liu, K. Wang, H. Wang, J. Liu, Y. Chen, J. Feng, C. Sha, X. Peng, and Y. Lou, "Evaluating Large Language Models in Class-Level Code Generation," in *IEEE/ACM Int'l Conf. on Software Engineering*, 2024, pp. 1–13.

[26] M. Chen, J. Tworek, H. Jun, Q. Yuan, Pinto, Henrique Ponde de Oliveira, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman *et al.*, "Evaluating Large Language Models Trained on Code," *arXiv:2107.03374*, 2021.

[27] W. Wang, Z. Gan, T. Li, K. Zhu, S. Fu, Z. Mao, J. Lou, Z. Ren, L. Hou, Z. Dai *et al.*, "A Survey on Retrieval-Augmented Text Generation," *arXiv:2402.19473*, 2024.

[28] D. Su, X. Ren, J. Zhang, J. Heng, R. Wei, Y. Yin, Z. Zhang, X. Yue, W. Li, J. Zhai *et al.*, "A Comprehensive Survey of Retrieval-Augmented Text Generation," *arXiv:2401.06800*, 2024.

[29] J. Chen, J. Yoon, S. Ebrahimi, S. Arik, T. Pfister, and S. Jha, "Self-Reflective Reasoning for Code Generation," *EMNLP Findings*, pp. 5190–5213, 2023.

[30] Z. Li and H. Wang, "Code Generation with Self-Critique," *ICSE*, pp. 842–853, 2024.

[31] K. Zhang, Z. Li, J. Li, G. Li, and Z. Jin, "Improving Code Generation via Self-Evaluation," *arXiv:2305.04087*, 2023.

[32] J. Kim, Y. Ahn, H. Cha, H. Lee, S. Lee, T. Kim, S. Yoon, S. Shin, and S. Oh, "Structured Reasoning in Large Language Models," *arXiv:2307.10123*, 2023.

[33] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models," *arXiv:2201.11903*, 2022.

[34] R. Singh, S. Kulal, N. Karande, and R. Chandra, "Automated Debugging and Verification in Code Generation," *ICSE*, pp. 1232–1243, 2023.

[35] J. Lee, M. Kim, J. Lee, and S. Ryu, "Syntax and Semantic Error Correction in Neural Code Generation," *AAAI*, pp. 6312–6320, 2023.

[36] I. Ince *et al.*, "Enhanced Evaluation Methods for Code Generation Models," 2024.

[37] W. Zheng *et al.*, "Comprehensive Benchmarking of Code Generation Models," 2024.

[38] M. Günther, J. Ong, I. Mohr, A. Abdessalem, T. Abel, M. K. Akram *et al.*, "Jina Embeddings 2: 8192-Token General-Purpose Text Embeddings for Long Documents," 2024.