The Cost of AI-Assisted Coding:
Energy vs. Accuracy in Language Models

Negar Alizadeh, Boris Belchev,
Nishant Saurabh, Patricia Kelbert,

BENEVOL 2025

Published in: IEEE/ACM International Conference on Mining Software Repositories (MSR) 2025

Introduction

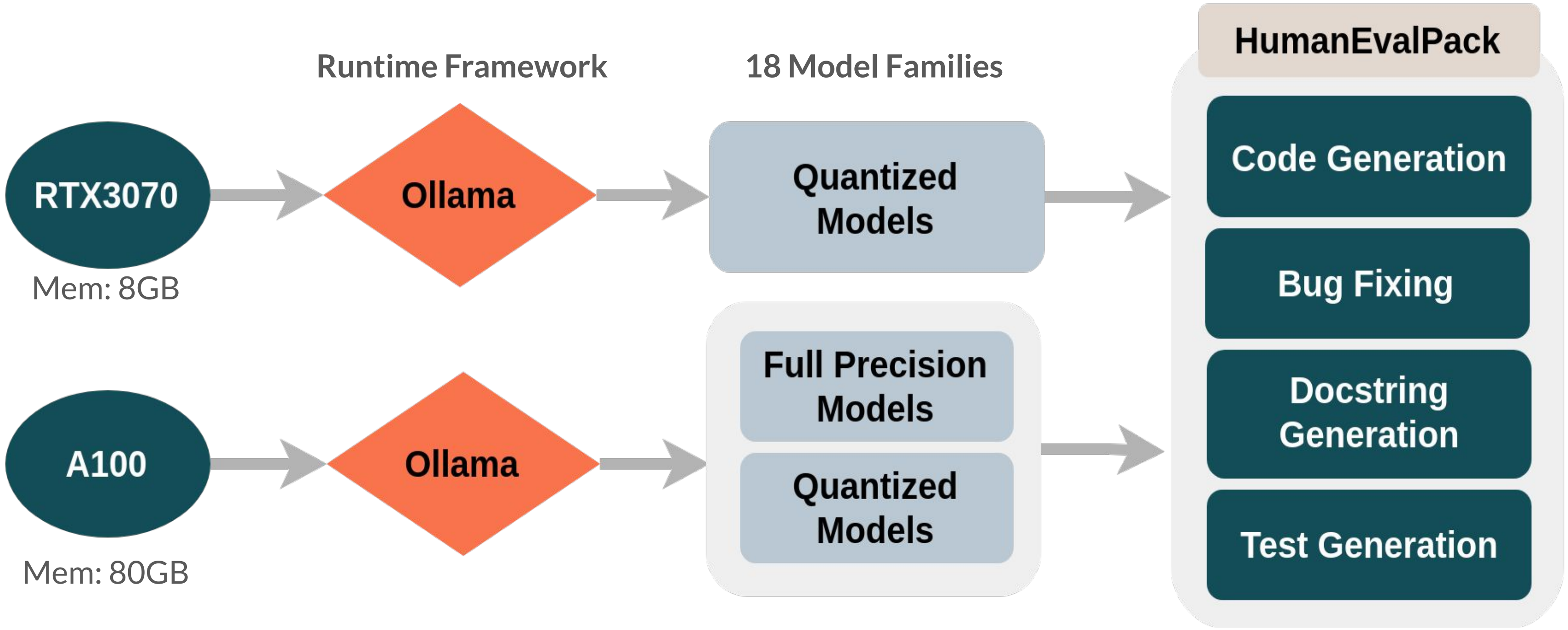
- Energy usage of **processes** and products of software development.
- With third-party APIs, there are data privacy issues, and cost concerns
    - Interest in locally deploying (open weight) language models.

- Challenges:
    - High energy consumption of LLMs
    - Difficulty running even modest-sized LLMs without a powerful GPU
    - Choosing the right model for your needs

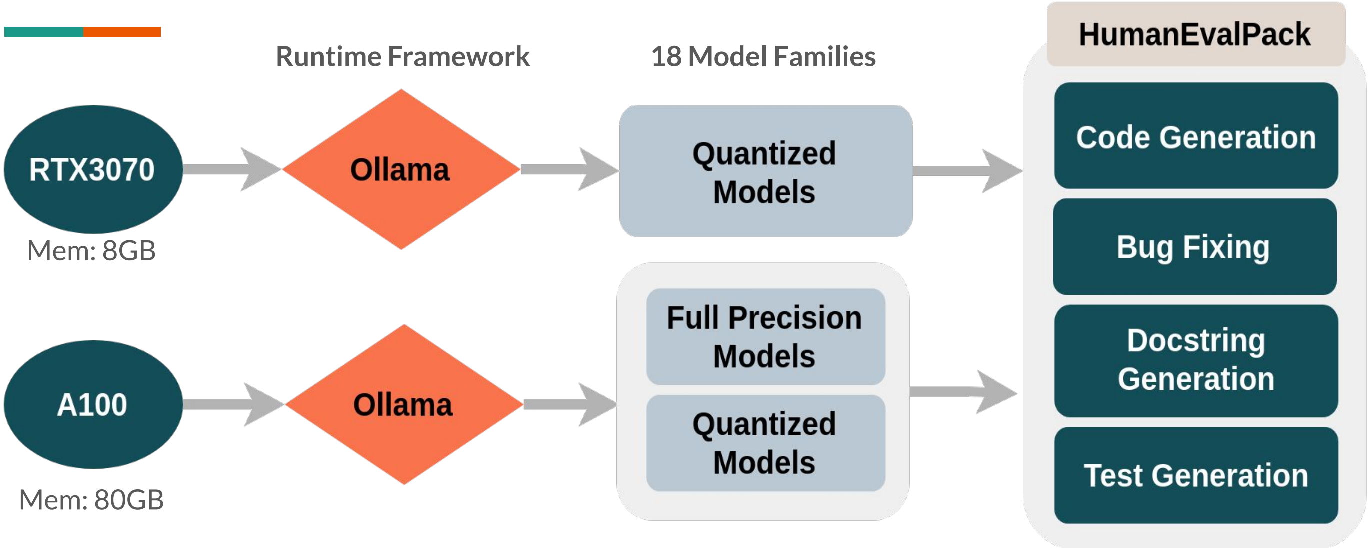
Goal

To investigate the energy consumption of (open weight) LLMs during **inference** in some software development tasks: code generation, bug fixing, docstring generation, and test case generation.

Methodology



Methodology



A100	**CPU:** 2 × AMD 7313 - 3GHz — Mem: 1TB — Cache: 32MB — Governer: Performance **GPU:** NVIDIA A100 PCIe — Mem: 80GB — PowerMizer: High Performance **OS:** AlmaLinux 8.10 (64-bit)

Methodology

- **Runtime framework:** Ollama
- **Models Evaluated:** 18 model families, general-purpose and code-specific (quantized and full-precision)

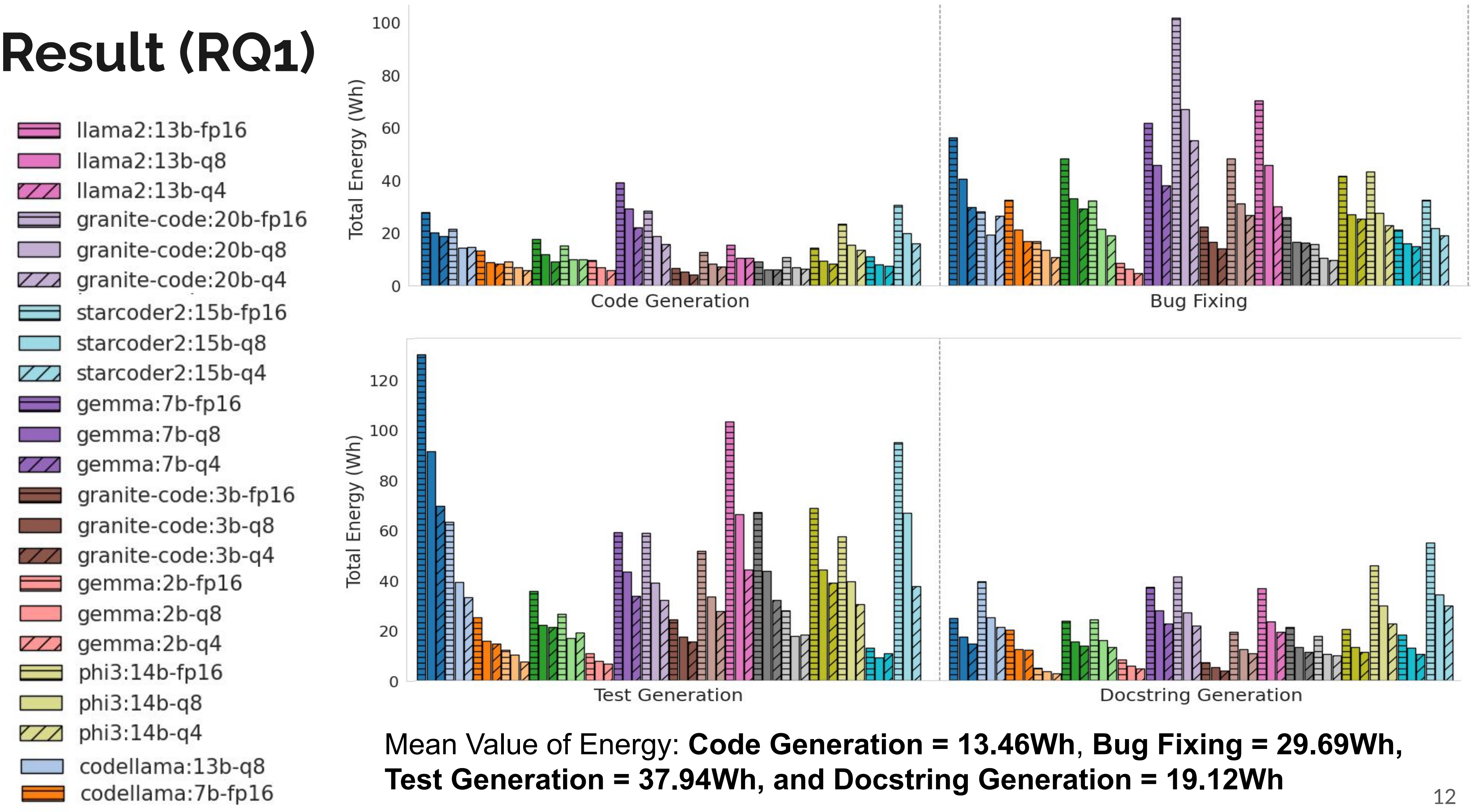
Code-Specific	General-Purpose
Codellama (7b-13b)	Llama2 (7b-13b)
Codegemma (7b)	Gemma (2b-7b)
Deepseek-coder (1.3b-6b)	Deepseek-llm (7b)
Starcoder2 (15b)	Llama3 (8b)
Granite-code (3b-8b-20b)	Mistral (7b)
Phi3 (3.8b-14b)	

Methodology

- **Accuracy:**
  - **Pass@1** for Code Generation, Bug Fixing, Docstring Generation
  - **Test Coverage and Test Correctness** for Test Generation
- **Hyperparameters:** temperature = 0.1, top-p = 0.95
- **Energy usage:** PyRAPL, PyNVML libraries for CPU and GPU
- **Energy:** Energy usage (Wh) and efficiency (tokens/J).

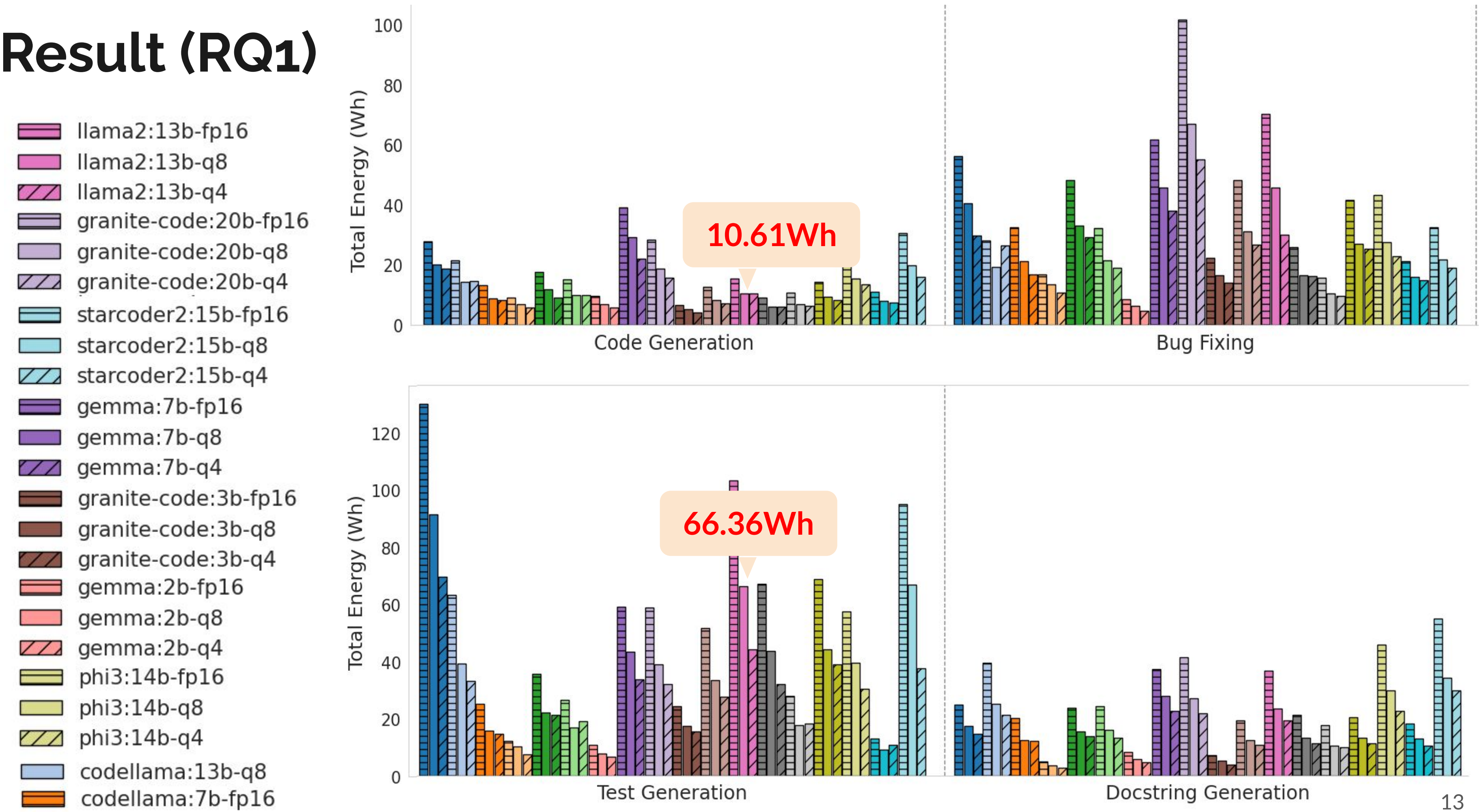
RQ1
Energy Usage Across Four Tasks

Result (RQ1)



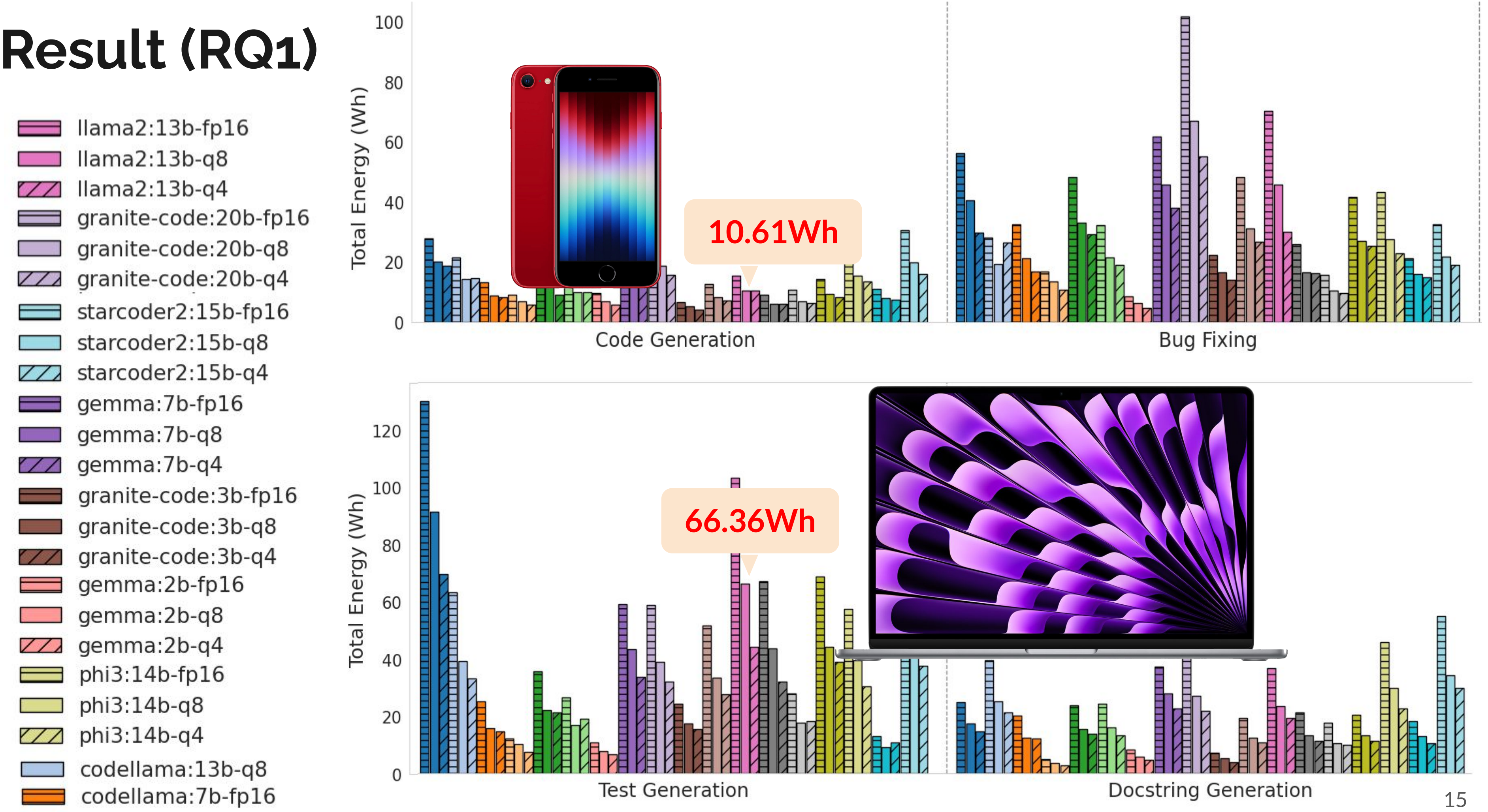
Mean Value of Energy: **Code Generation = 13.46Wh**, **Bug Fixing = 29.69Wh, Test Generation = 37.94Wh, and Docstring Generation = 19.12Wh**

Result (RQ1)



llama2:13b-fp16
llama2:13b-q8
llama2:13b-q4
granite-code:20b-fp16
granite-code:20b-q8
granite-code:20b-q4
starcoder2:15b-fp16
starcoder2:15b-q8
starcoder2:15b-q4
gemma:7b-fp16
gemma:7b-q8
gemma:7b-q4
granite-code:3b-fp16
granite-code:3b-q8
granite-code:3b-q4
gemma:2b-fp16
gemma:2b-q8
gemma:2b-q4
phi3:14b-fp16
phi3:14b-q8
phi3:14b-q4
codellama:13b-q8
codellama:7b-fp16
Total Energy (Wh)
10.61Wh
Code Generation
Bug Fixing
66.36Wh
Test Generation
Docstring Generation

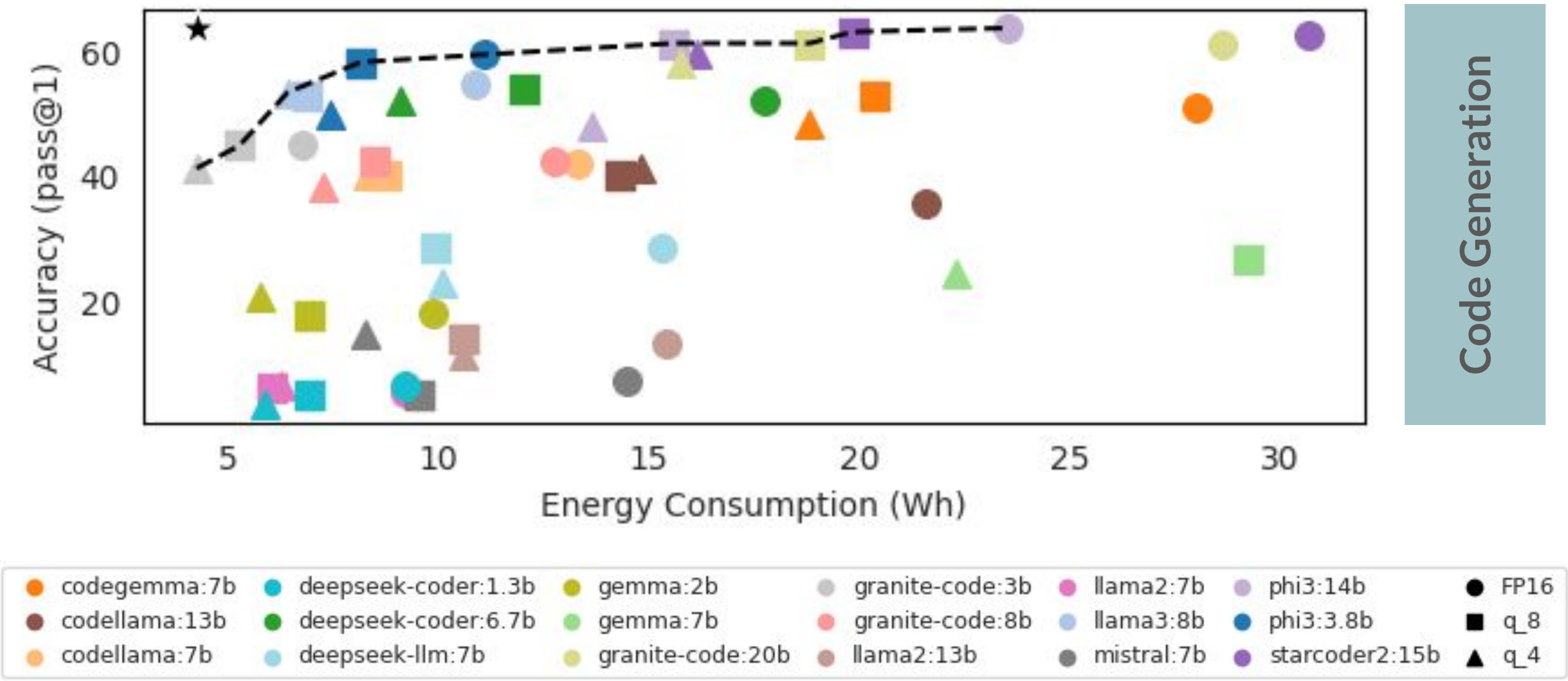
Result (RQ1)



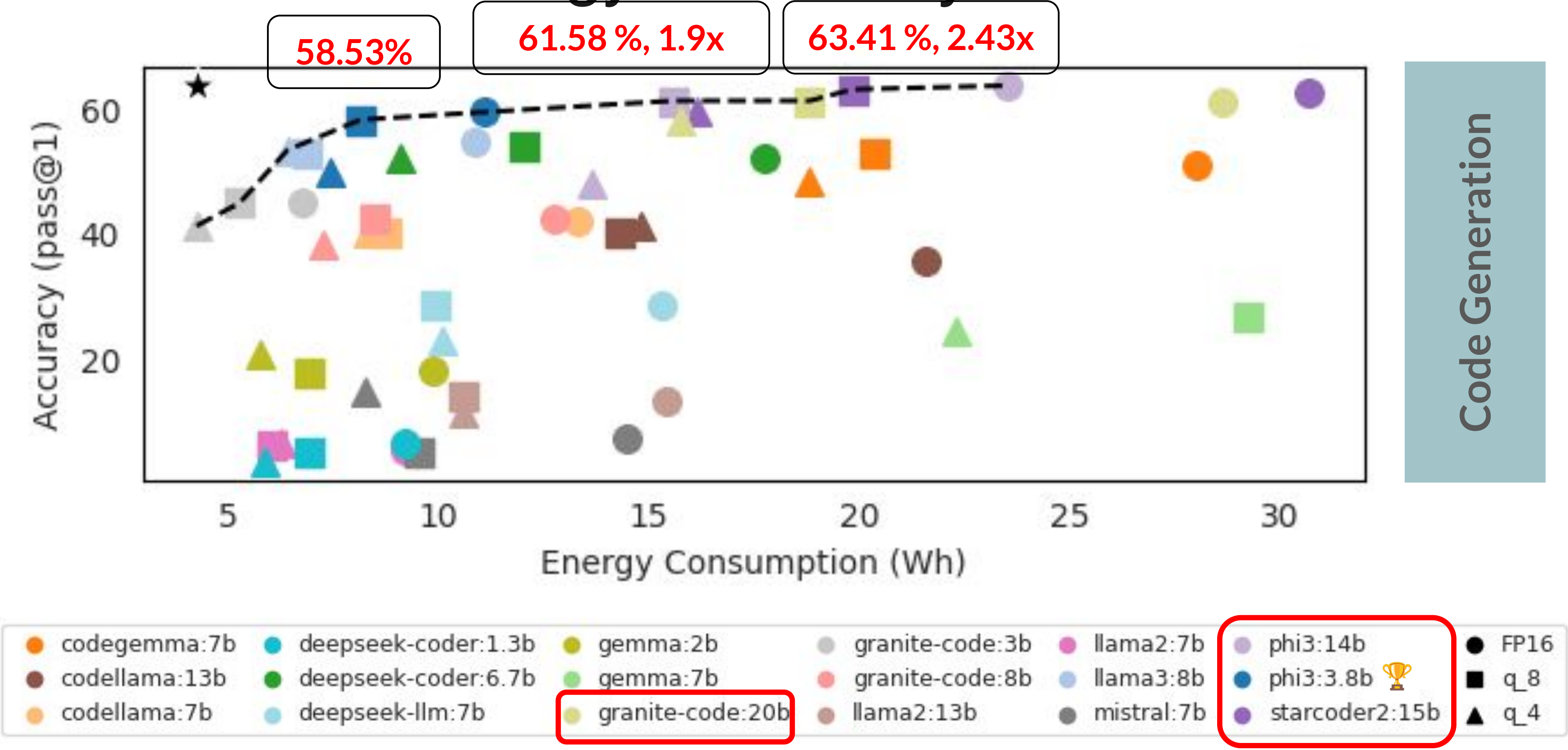
llama2:13b-fp16
llama2:13b-q8
llama2:13b-q4
granite-code:20b-fp16
granite-code:20b-q8
granite-code:20b-q4
starcoder2:15b-fp16
starcoder2:15b-q8
starcoder2:15b-q4
gemma:7b-fp16
gemma:7b-q8
gemma:7b-q4
granite-code:3b-fp16
granite-code:3b-q8
granite-code:3b-q4
gemma:2b-fp16
gemma:2b-q8
gemma:2b-q4
phi3:14b-fp16
phi3:14b-q8
phi3:14b-q4
codellama:13b-q8
codellama:7b-fp16
Total Energy (Wh)
0
20
40
60
80
100
10.61Wh
Code Generation
Bug Fixing
Total Energy (Wh)
0
20
40
60
80
100
120
66.36Wh
Test Generation
Docstring Generation

RQ2
Energy vs. Accuracy Trade-Offs

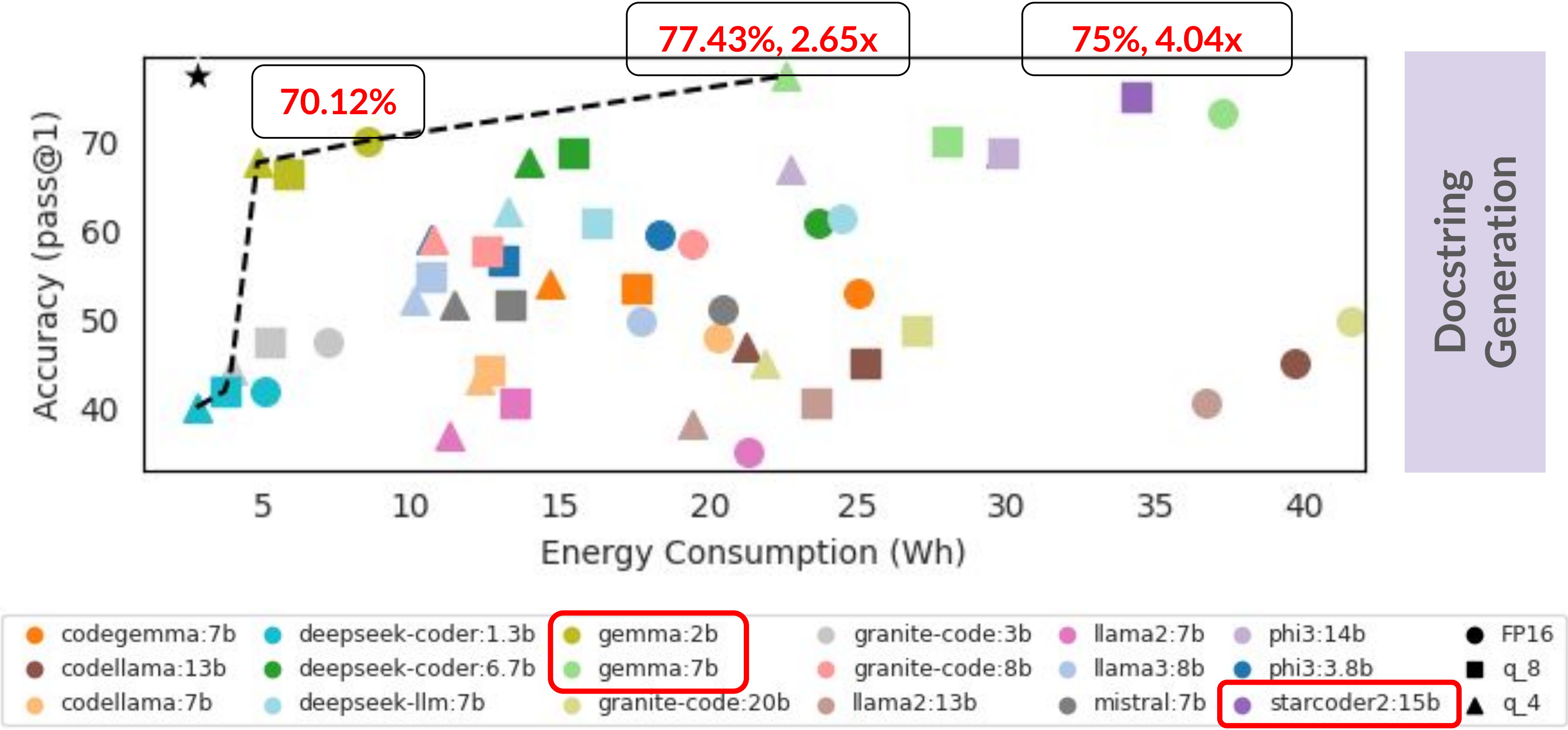
Result (RQ2: Energy vs. Accuracy Trade-Offs)



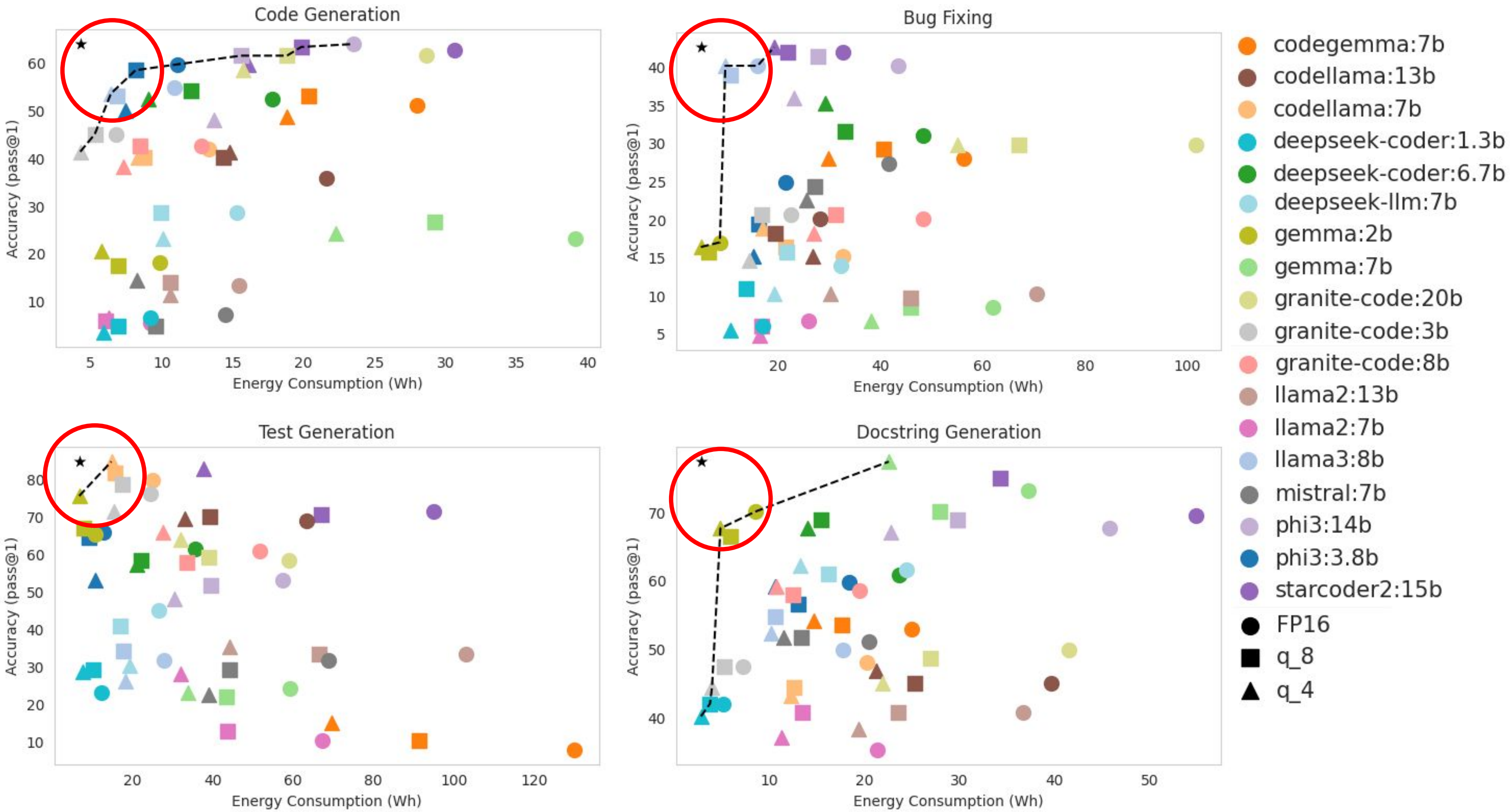
Result (RQ2: Energy vs. Accuracy Trade-Offs)



Result (RQ2: Energy vs. Accuracy Trade-Offs)



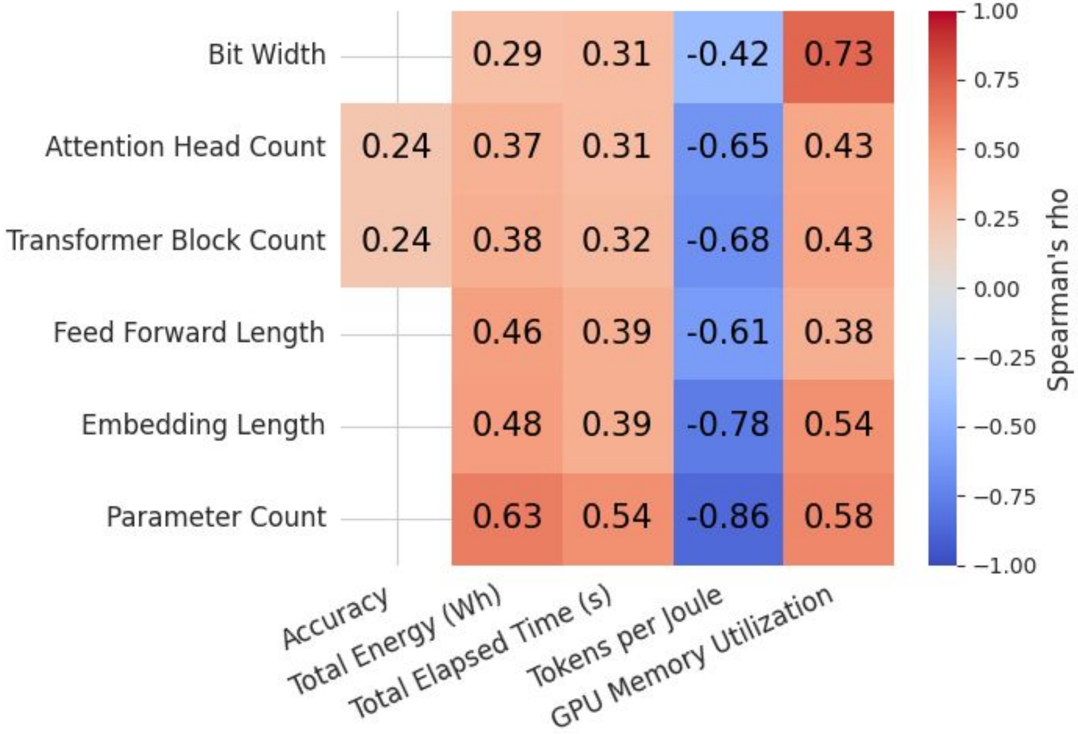
For Energy and Accuracy, it’s not necessarily a trade-off



RQ3
Model Characteristics

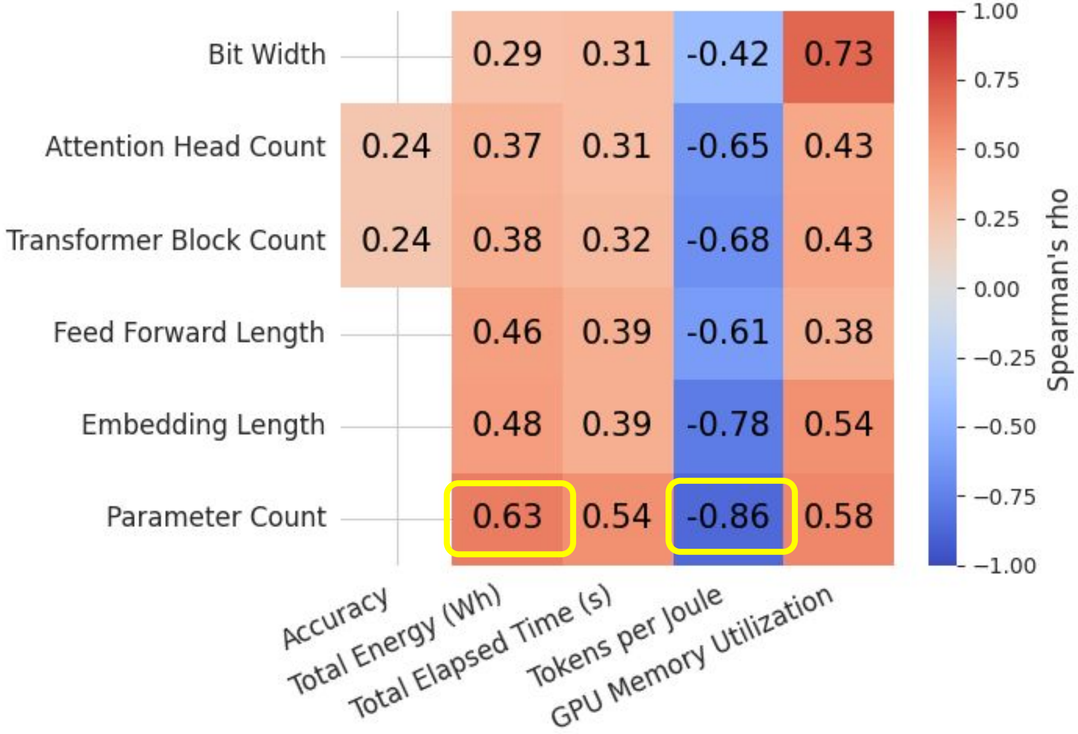
Result (RQ3: Model Characteristics)

Spearman’s correlation matrix for all models across all tasks on GPU A100
($p - \text{value} < 0.0016$)



Result (RQ3: Model Characteristics)

Models with larger number of parameters need more energy to generate an output token.



Result (RQ3: Model Characteristics)

But they do not necessarily produce more accurate results.

RQ4
Code-Specific LLMs vs. General-Purpose LLMs

Result (RQ4: Code-Specific vs. General-Purpose)

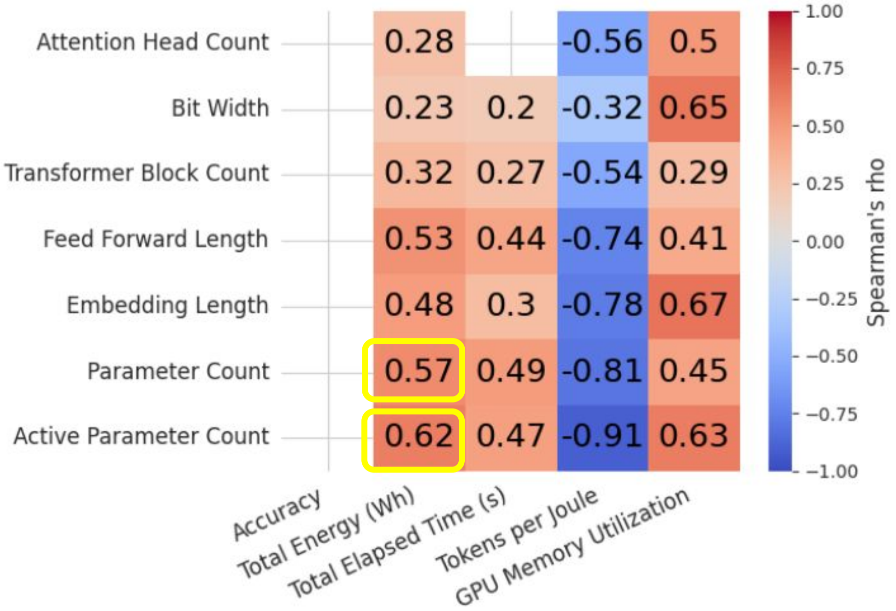
- **Excluding energy usage**, coding-specific LLMs exhibit better mean accuracy than general-purpose LLMs
- **Considering energy usage**, general models appear among pareto frontiers.

Coding models should be designed to be both accurate and energy-efficient.

Further Analysis

Parameter count matters for energy efficiency

Active parameter count matters more



$$E = \beta_0 + \beta_1 P + \beta_2 O + \beta_3 W + \dots + \beta_4 I + \varepsilon$$

Further Analysis

$$E = \beta_0 + \beta_1 P + \beta_2 O + \beta_3 W + \ldots + \beta_4 I + \varepsilon$$

Active Parameters

Output Length

Bit Width

Further Analysis

- Active **P**arameter count, **O**utput tokens, and bit **W**idth combined have high explanatory power for **Energy usage**
- Weights vary per task

Code Generation					
Predictor	**Coefficient (β)**	**SE**	**p-value**	**R^2**	**Adj. R^2**
Intercept	7.5049	0.600	<0.001		
Input Tokens_{std}	-0.3033	0.462	0.511		
Output Tokens_{std}	4.6003	1.695	0.007	0.743	0.729
Parameter Count_{std}	5.2472	0.551	<0.001		
Quantization Level	3.4057	0.429	<0.001		
Bug Fixing					
Predictor	**Coefficient (β)**	**SE**	**p-value**	**R^2**	**Adj. R^2**
Intercept	16.6326	1.348	<0.001		
Input Tokens_{std}	-2.3272	1.069	0.029		
Output Tokens_{std}	9.2888	2.429	<0.001	0.798	0.786
Parameter Count_{std}	13.2138	1.456	<0.001		
Quantization Level	7.7707	1.070	<0.001		
Docstring Generation					
Predictor	**Coefficient (β)**	**SE**	**p-value**	**R^2**	**Adj. R^2**
Intercept	9.9995	0.826	<0.001		
Input Tokens_{std}	-0.0611	0.574	0.915		
Output Tokens_{std}	3.6721	0.656	<0.001	0.900	0.895
Parameter Count_{std}	7.3851	0.628	<0.001		
Quantization Level	5.1572	0.669	<0.001		
Test Generation					
Predictor	**Coefficient (β)**	**SE**	**p-value**	**R^2**	**Adj. R^2**
Intercept	20.5233	1.779	<0.001		
Input Tokens_{std}	0.7161	1.316	0.586		
Output Tokens_{std}	14.8815	1.440	<0.001	0.880	0.873
Parameter Count_{std}	13.9443	1.790	<0.001		
Quantization Level	10.4284	1.351	<0.001		

Main Takeaways

On the energy efficiency of LLMs in four software development tasks

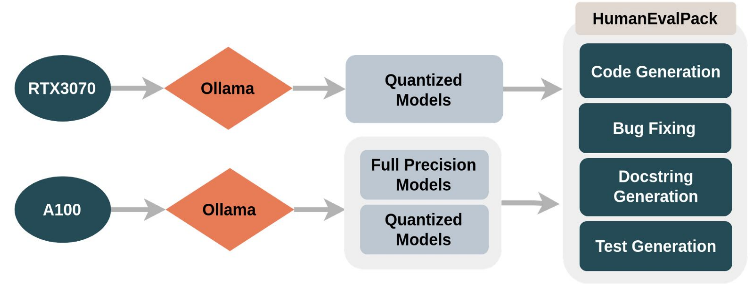
1. For energy and accuracy, it does not need to be a trade-off

2. (Active) parameter count has a strong connection to energy efficiency. Not so much to accuracy

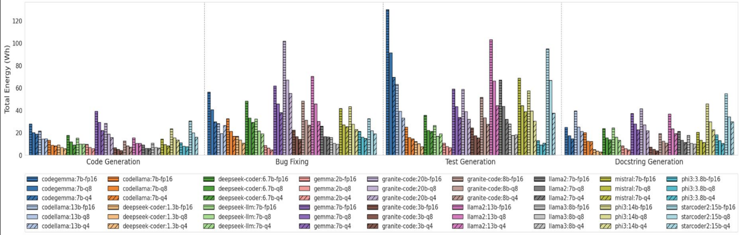
3. The combination of Parameter count, Output length and bit Width is a good predictor for energy usage

Thank You!

Methodology

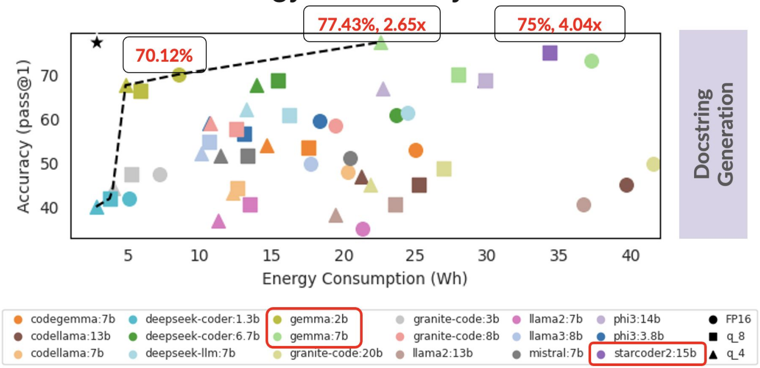


Result (RQ1: Across Task Energy Usage) - A100



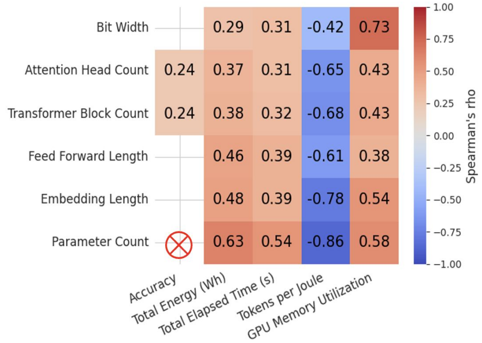
The mean energy consumed across all the models for Code Generation, Docstring Generation, Bug Fixing, and Test Generation was, respectively, 13.46Wh, 19.12Wh, 29.69Wh, and 37.94Wh

Result (RQ2: Energy vs. Accuracy Trade-Offs)



Result (RQ3: Model Characteristics)

Larger models do not necessarily produce more accurate results than smaller ones.



Negar Alizadeh (n.s.alizadeh@uu.nl)