Knowledge Transfer for Code Intelligence: PEFT and LLM-based Agents

Fatemeh H. Fard

Assistant professor, University of British Columbia

Low Resource Languages and Scientific Programming Languages

Computational Efficiency

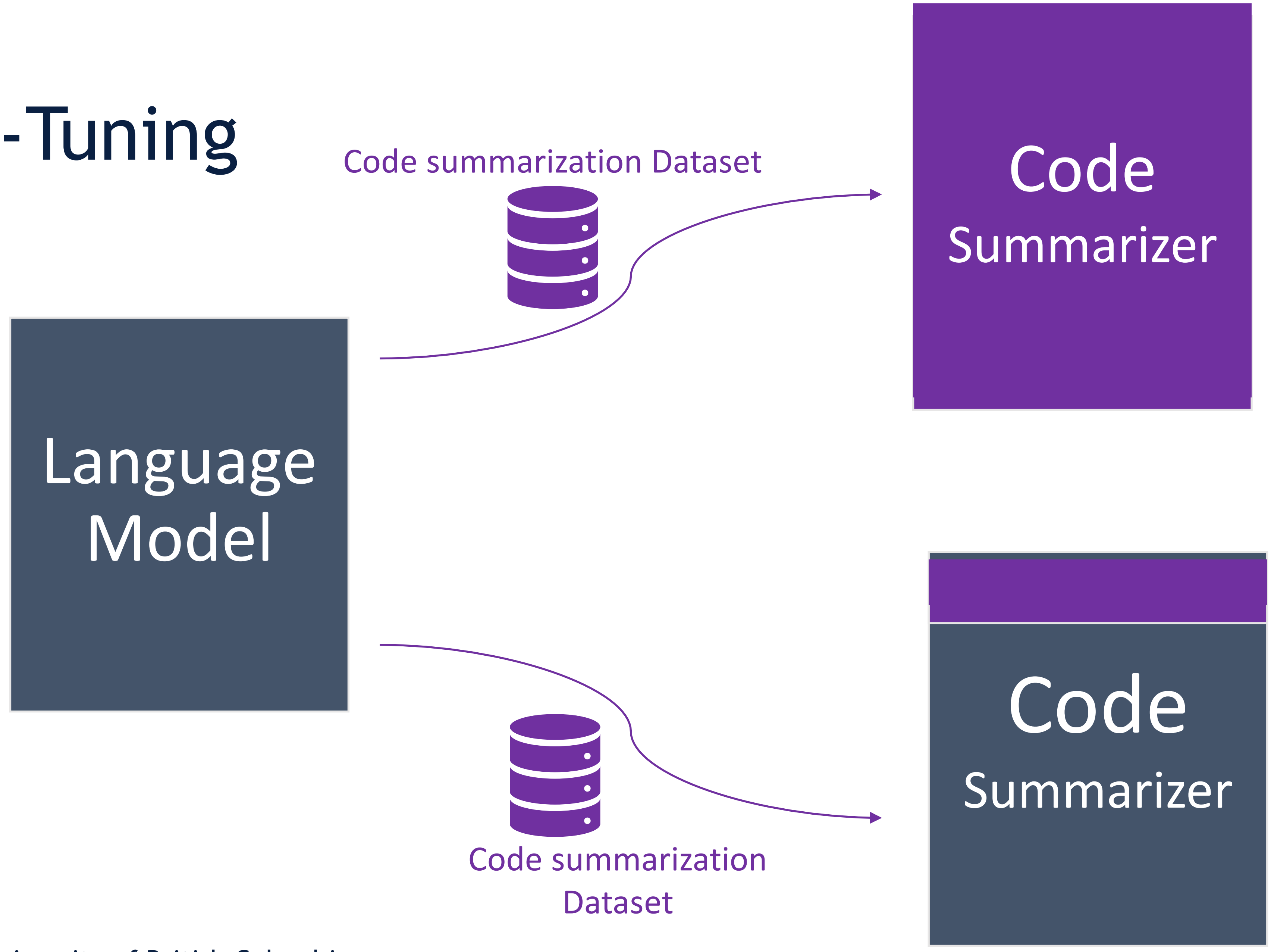
- Low resource languages
- Noise label detection
- Knowledge transfer
- RAG-based LLMs
- AIWare and agent-based LLMs

- Computational efficiency
- Performance increase
- Adding new knowledge to LLMs
- Automating the pipeline

Knowledge Transfer for Software Engineering

- Using PL-LMs and Parameter Efficient Fine Tuning (PEFT)
- Using (NL)-LLM-based Agents and Stack Overflow

Fine-Tuning


Code summarization Dataset
Code
Summarizer
Language
Model
Code
Summarizer
Code summarization
Dataset

PEFT Categories

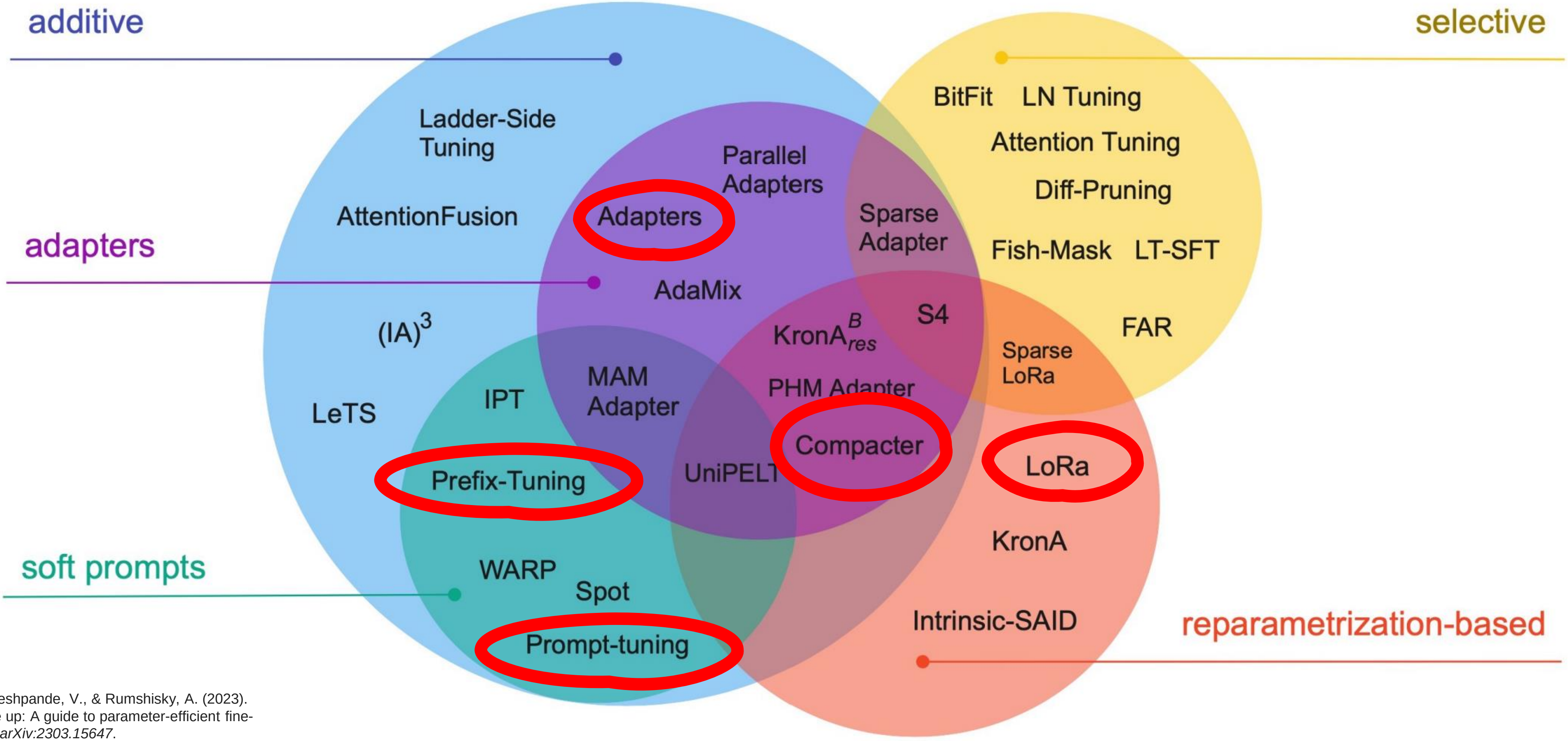


Fig from: Lialin, V., Deshpande, V., & Rumshisky, A. (2023). Scaling down to scale up: A guide to parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.15647*.

PEFT Benefits

Less storage requirements

Comparable results

Computational efficiency

Store only updated parameters

Specially for low resource languages

(sometimes) memory and processing

SE Specific Aspects

- Bimodal knowledge transfer
- SE specific adapters for knowledge transfer from multiple programming languages

Divyam Goel

Ramansh Grover

Iman Saberi

30th IEEE/ACM International Conference on Program Comprehension

On The Cross-Modal Transfer from Natural Language to Code through Adapter Modules

Divyam Goel*
Indian Institute of Technology
Roorkee, India
dgoel@bt.iitr.ac.in

Ramansh Grover*
Delhi Technological University
Delhi, India
ramanshgrover_2k18co281@dtu.ac.in

Fatemeh H. Fard
University of British Columbia
Canada
fatemeh.fard@ubc.ca

- Focus on bimodal knowledge transfer from NL to PL
- Cloze test and Code clone detection
- RoBERTa, MODE-X, CODEBERT

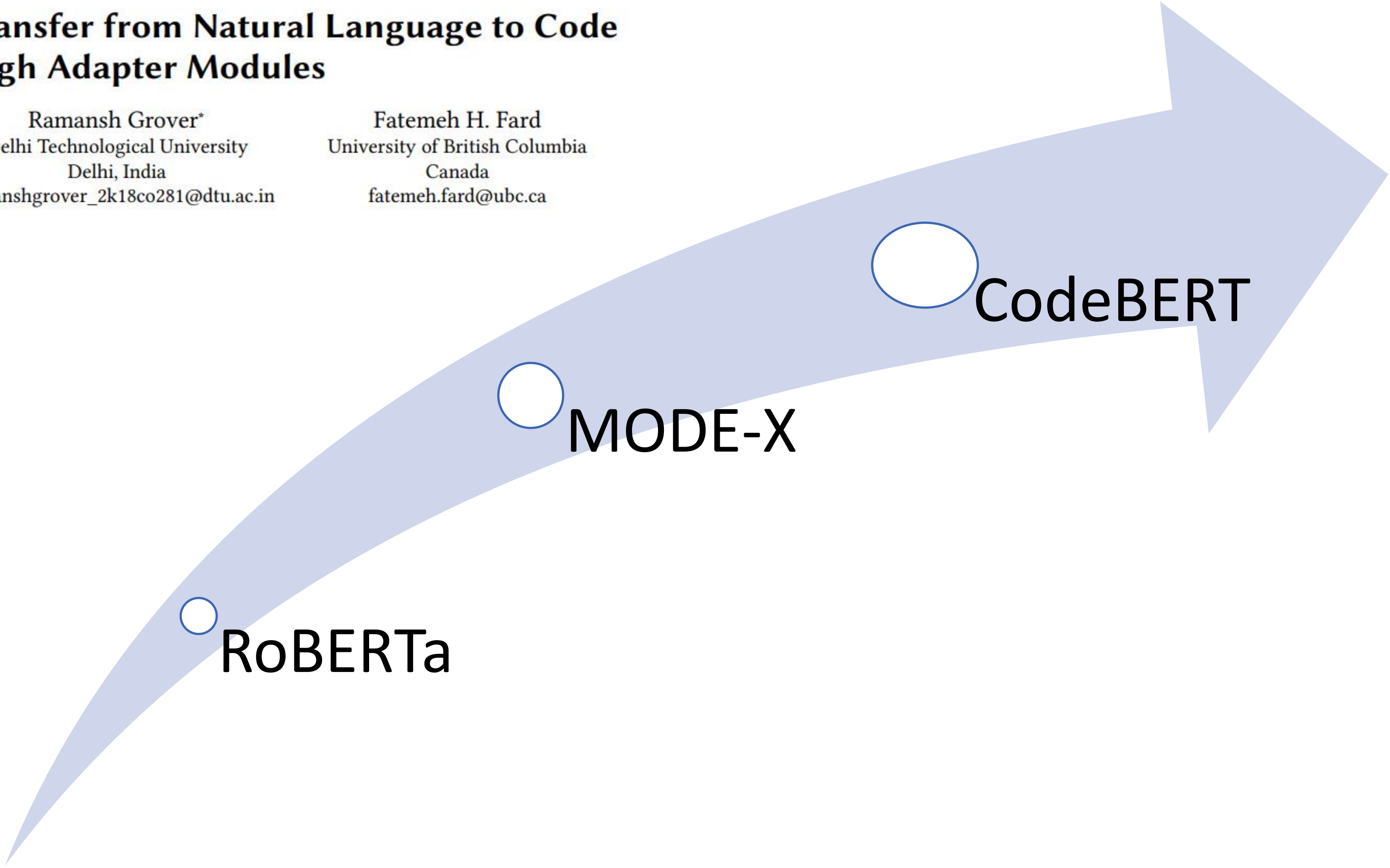
30th IEEE/ACM International Conference on Program Comprehension

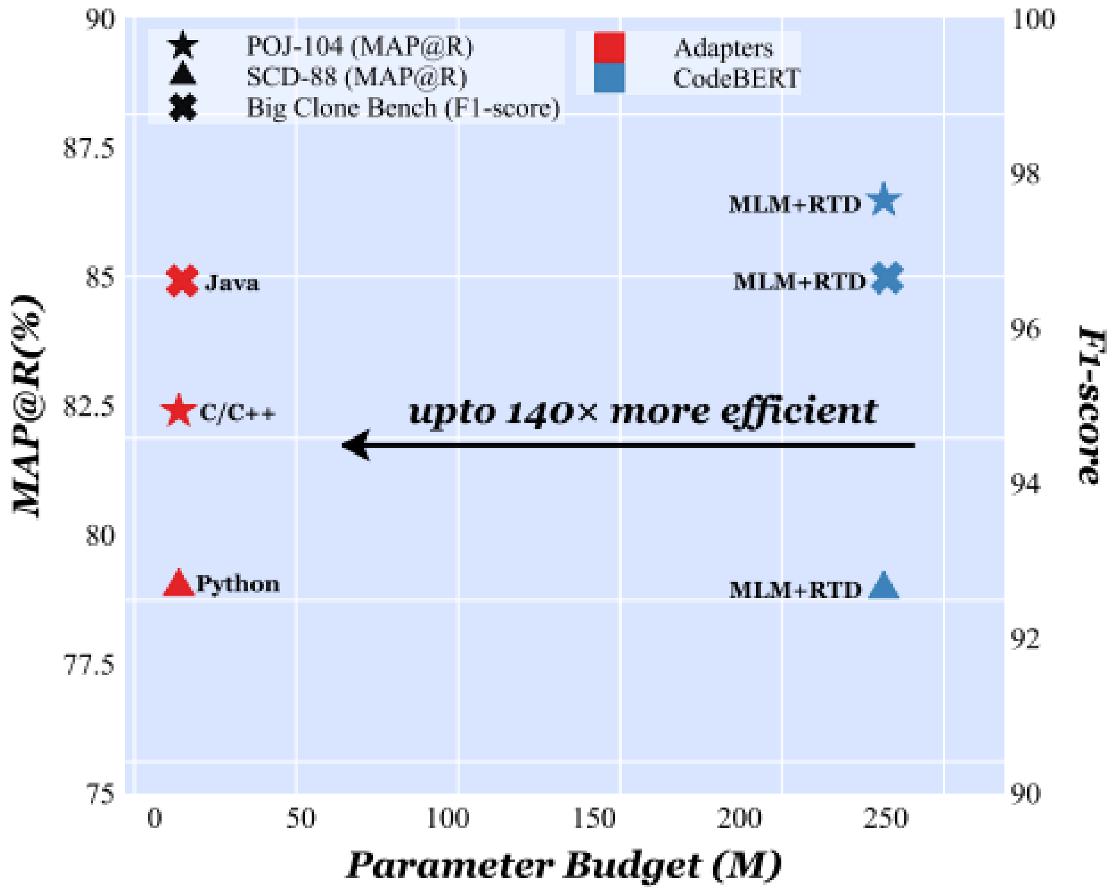
On The Cross-Modal Transfer from Natural Language to Code through Adapter Modules

Divyam Goel*
Indian Institute of Technology
Roorkee, India
dgoel@bt.iitr.ac.in

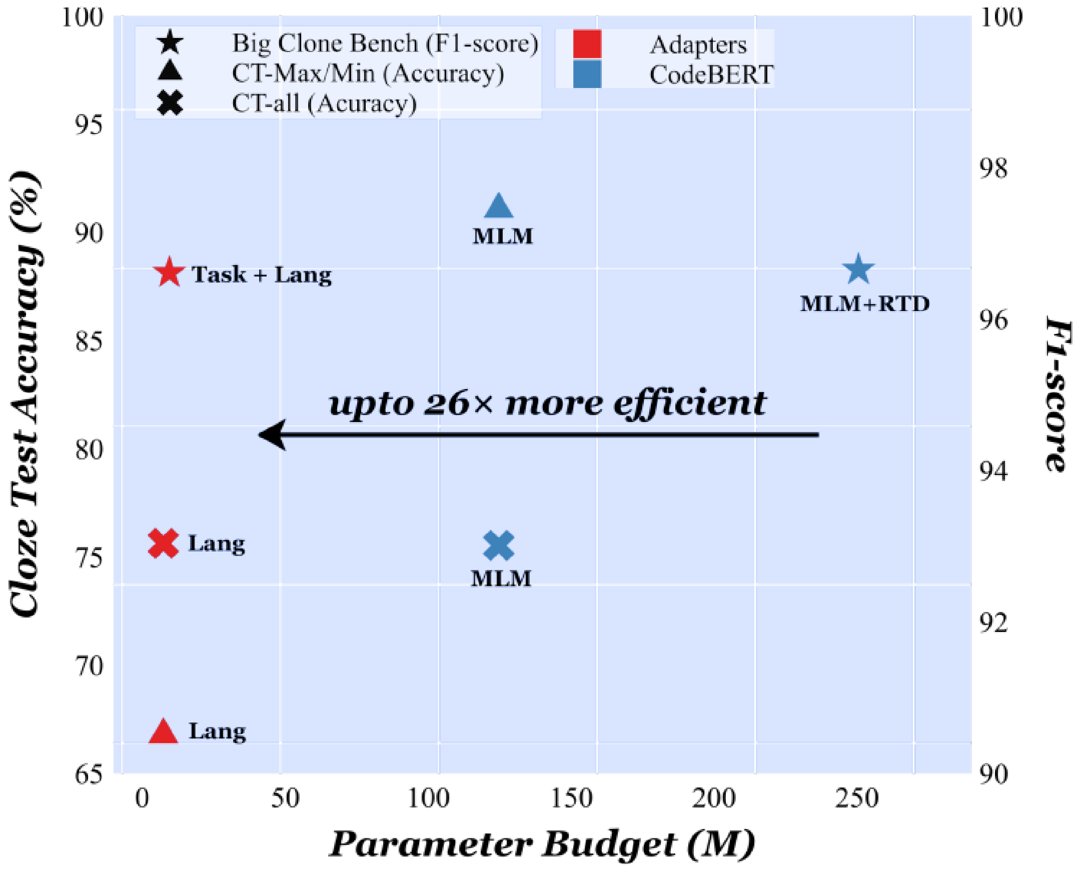
Ramansh Grover*
Delhi Technological University
Delhi, India
ramanshgrover_2k18co281@dtu.ac.in

Fatemeh H. Fard
University of British Columbia
Canada
fatemeh.fard@ubc.ca





Parameter budget of adapters and CodeBERT for code clone detection



Parameter budget of Java-adapters and CodeBERT in millions

Empirical Software Engineering (EMSE) 2024

Utilization of Pre-trained Language Models for Adapter-based Knowledge Transfer in Software Engineering

Iman Saberi · Fatemeh Fard · Fuxiang Chen

- Focus on low resource languages
- Code summarization
- RoBERTa, MODE-X, CODEBERT, GraphCodeBERT

Code Summarization Results

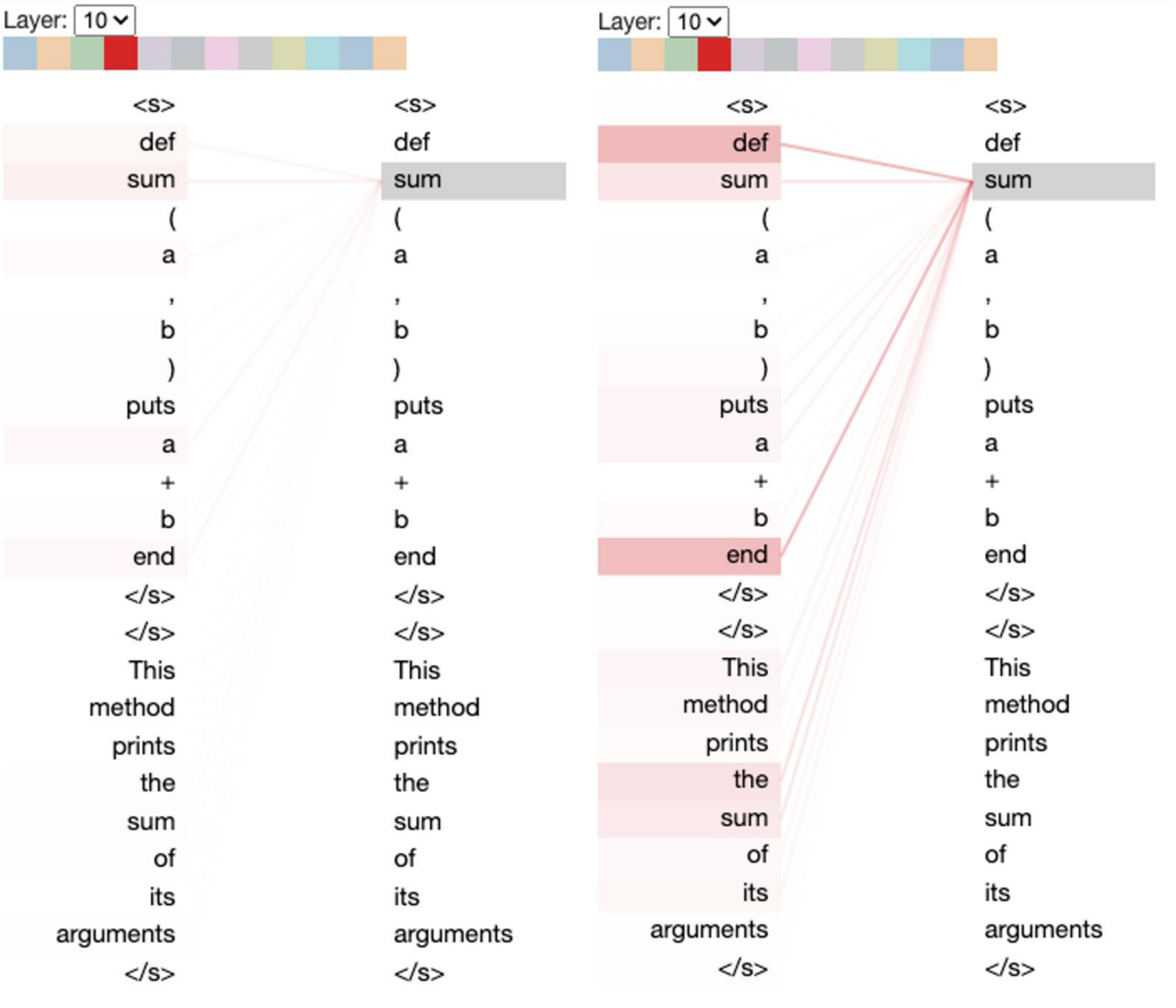
Smoot BLEU-4

Models/Languages	Ruby	JS	Go	Python	Java	PHP
GraphCodeBERT + TA	**14.53**	**16.54**	**23.74**	<u>18.73</u>	<u>19.08</u>	25.05
CodeBERT+TA	<u>14.12</u>	<u>15.67</u>	<u>23.21</u>	18.47	18.99	**25.55**
MODE-X	12.79	14.20	23.05	17.72	18.43	24.27
GraphCodeBERT	12.62	14.79	18.40	18.02	**19.22**	<u>25.45</u>
CodeBERT	12.16	14.90	18.07	**19.06**	17.65	25.16
RoBERTa	11.17	11.90	17.72	18.14	16.47	24.02

MODE-X has better or on par results with Code-LMs

Adapters outperform FFT for low resource languages

Ruby Attention



Beyond Empirical Studies

Software Engineering-Specific PEFT Methods

CODEBERTER

Iman Saberi, Fatemeh H. Fard, Model-Agnostic Syntactical Information for Pre-Trained Programming Language Models.

MSR 2023

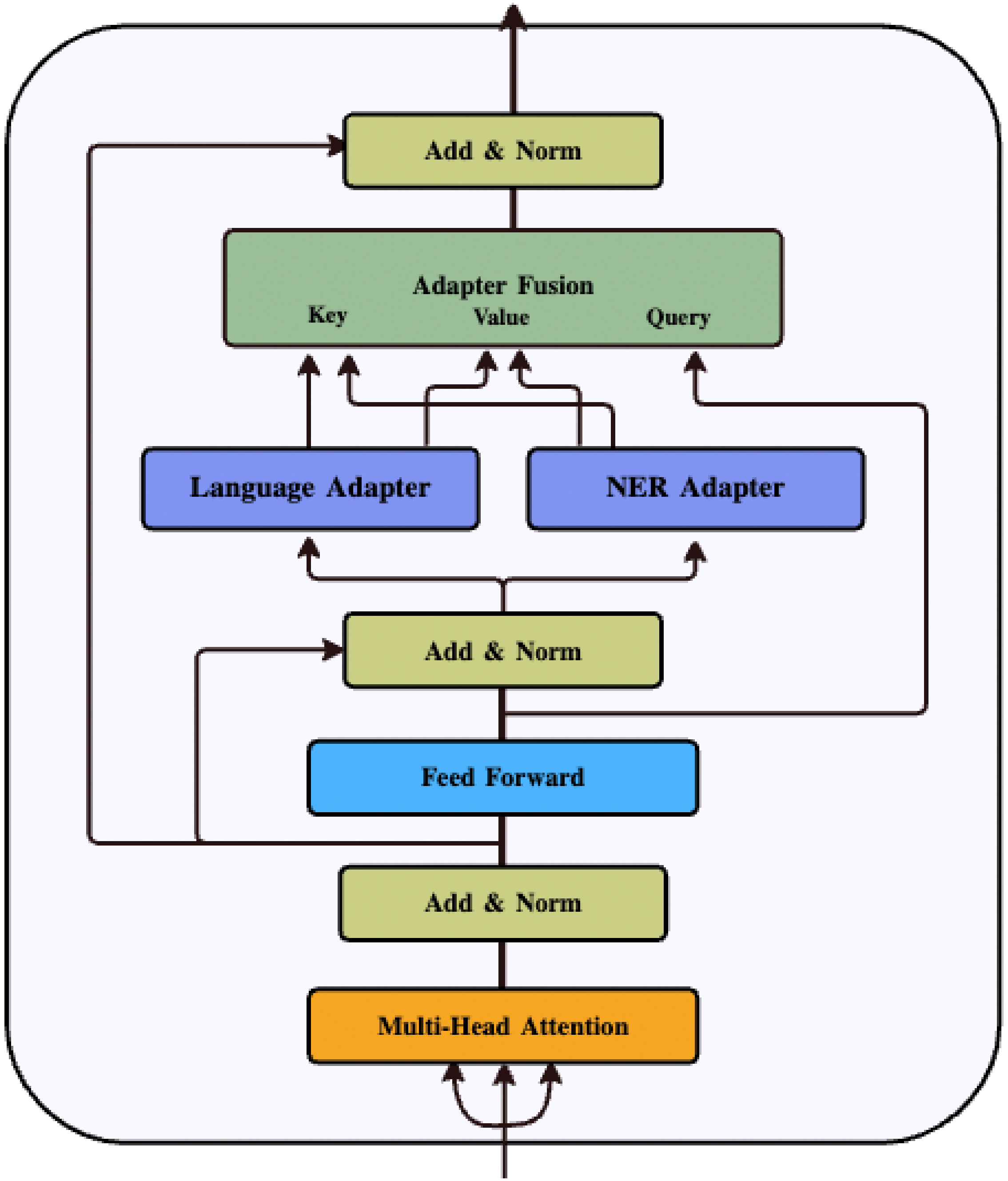
Avoid pre-training while adding new information

Computational Efficiency + Avoid Pre-training in Imposing Code Structure = NER Adapter

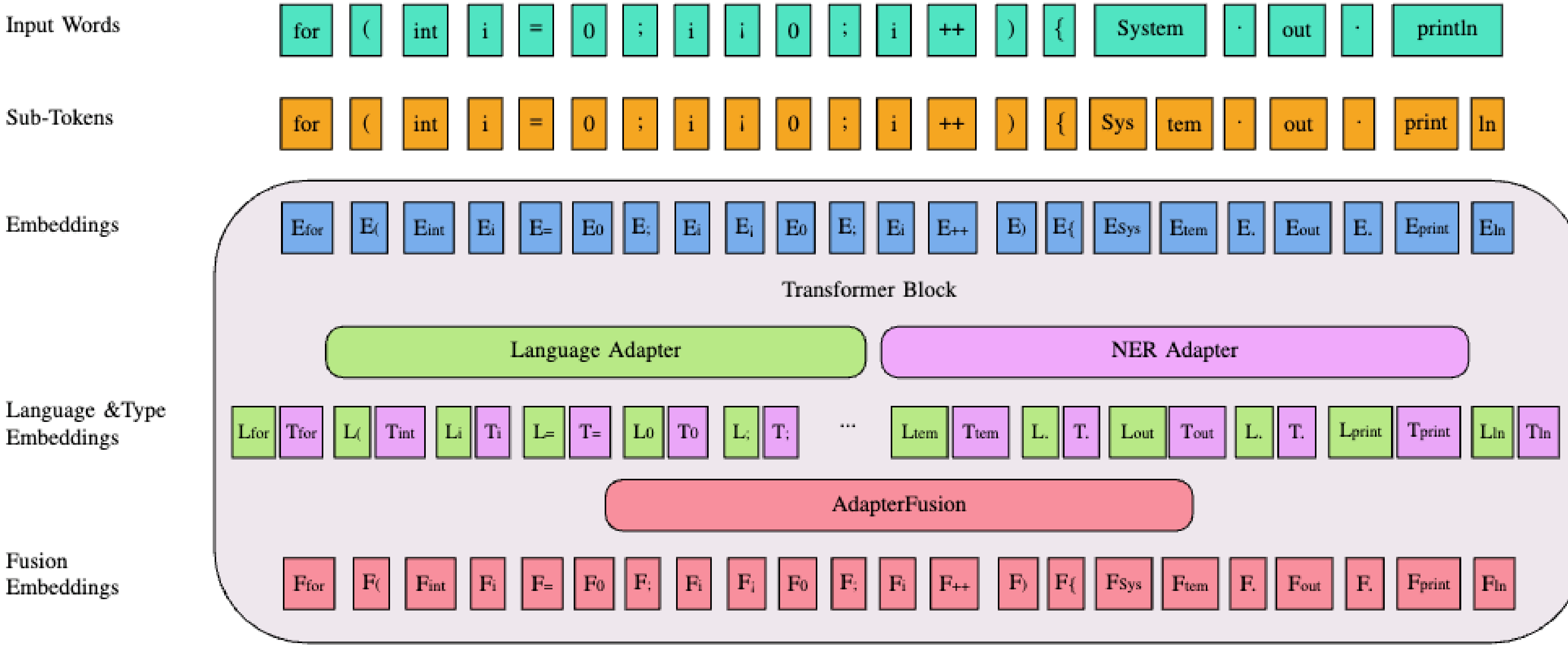
NER Adapter

Token Type

Classification Loss (TTC)



Overall Architecture



The input data flow for the sample when fed into a transformer block equipped with NER, language and Fusion adapters.

Code Summarization

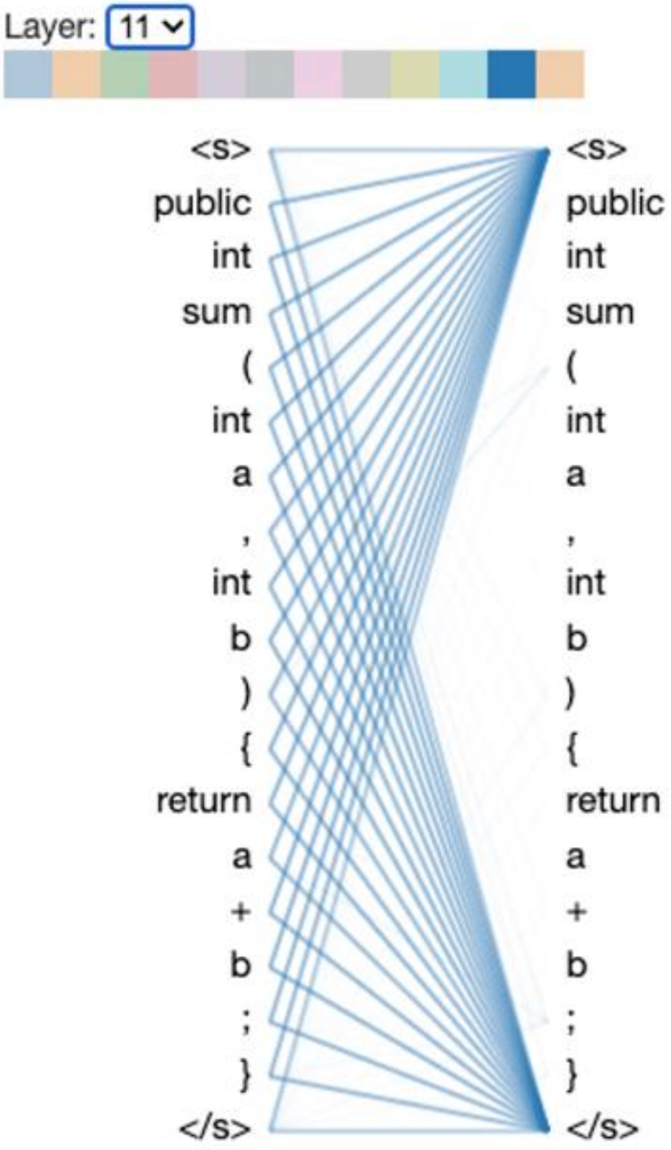
Automatically generating descriptions of the functionality of a given code

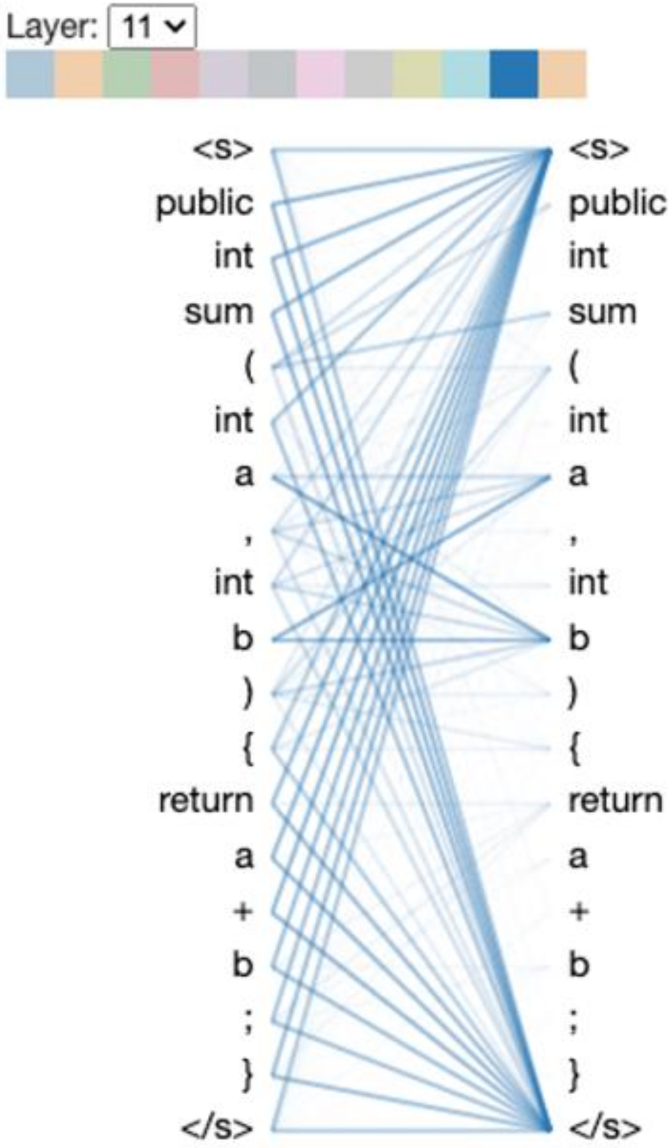
Models	Ruby	JavaScript	Go	Python	Java	Average
CodeBERTER	**15.90**	**16.12**	**23.34**	18.38	19.95	**18.738**

Models	Ruby	JavaScript	Go	Python	Java	Average
CodeBERTER	**15.90**	**16.12**	**23.34**	18.38	19.95	**18.738**
*polyglot*GraphCodeBERT [32]	14.95	15.79	18.92	18.90	19.91	17.694
*polyglot*CodeBERT [32]	14.75	15.80	18.77	18.71	20.11	17.48
DistillCodeT5	15.75	16.42	20.21	**20.59**	**20.51**	18.696
CodeT5 [3]	15.69	16.24	19.76	20.36	20.46	18.502
ProphetNet-Code [36]	14.37	16.60	18.43	17.87	19.39	17.332
CoTexT [36]	14.02	14.96	18.86	19.73	19.06	17.326
PLBART [12]	14.11	15.56	18.91	19.30	18.45	17.22
GraphCodeBERT	12.62	14.79	18.40	18.02	19.22	16.61
CodeBERT	12.16	14.90	18.07	19.06	17.65	16.36
RoBERTa [37]	11.17	11.90	17.72	18.14	16.47	15.08
Transformer [13]	11.18	11.59	16.38	15.81	16.26	14.24
seq2seq [38]	9.64	10.21	13.98	15.93	15.09	12.97

Attention Change with NER Adapter

- **CodeBERTER** (right)
- CodeBERT (left)

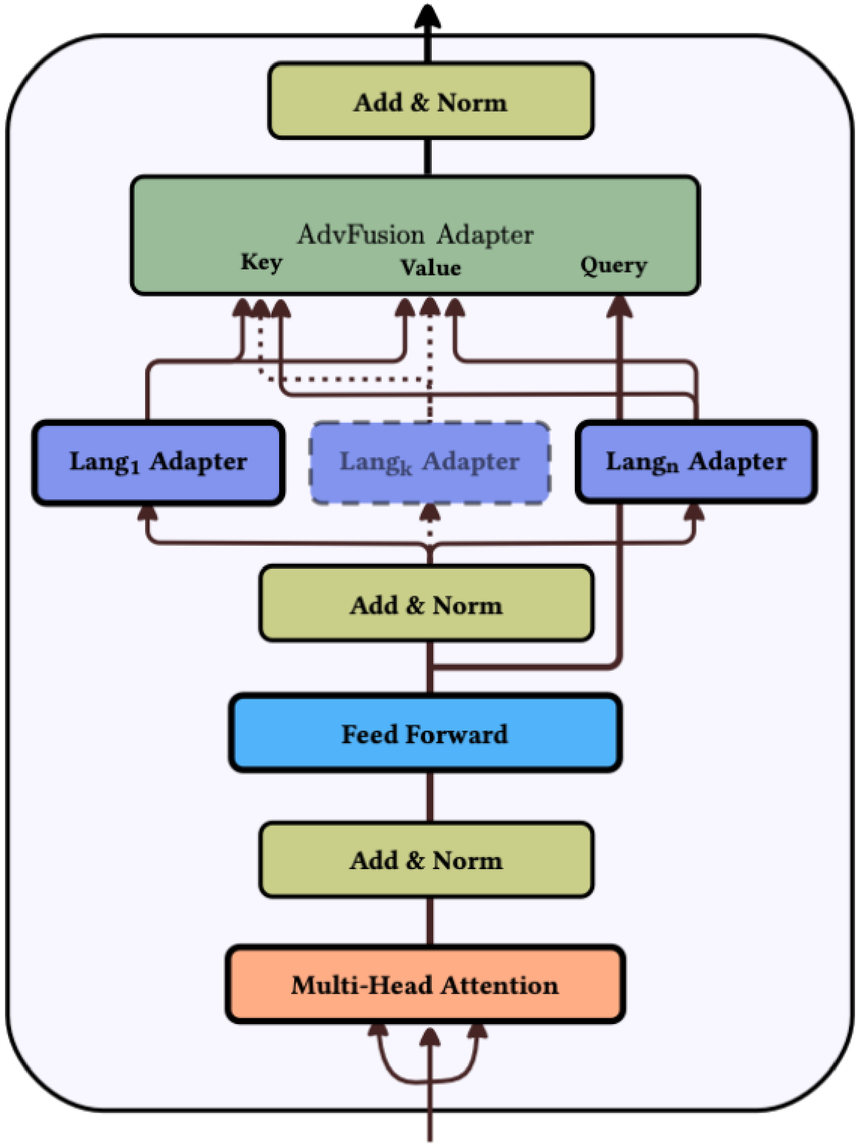


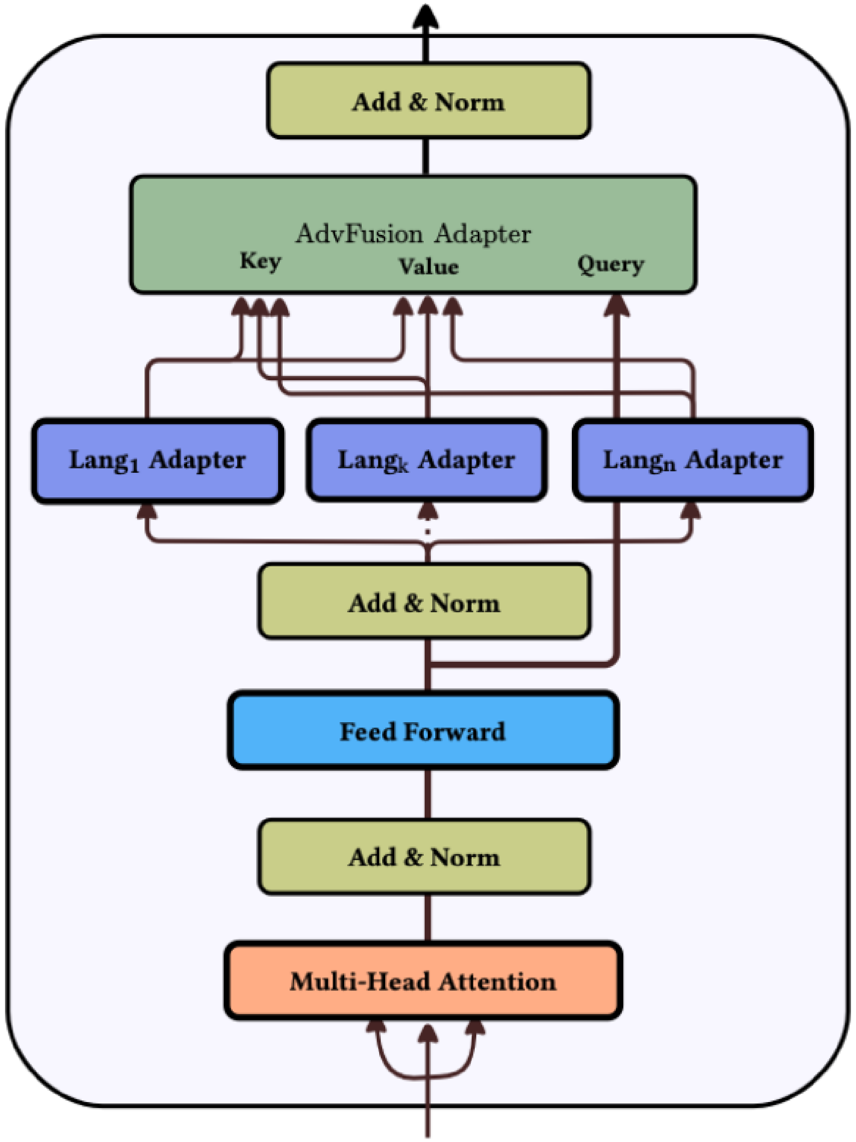


ADVFUSION

Iman Saberi, Fatemeh H. Fard, Fuxiang Chen,
AdvFusion: Multilingual Adapter-based Knowledge Transfer for Code Summarization

AdvFusion

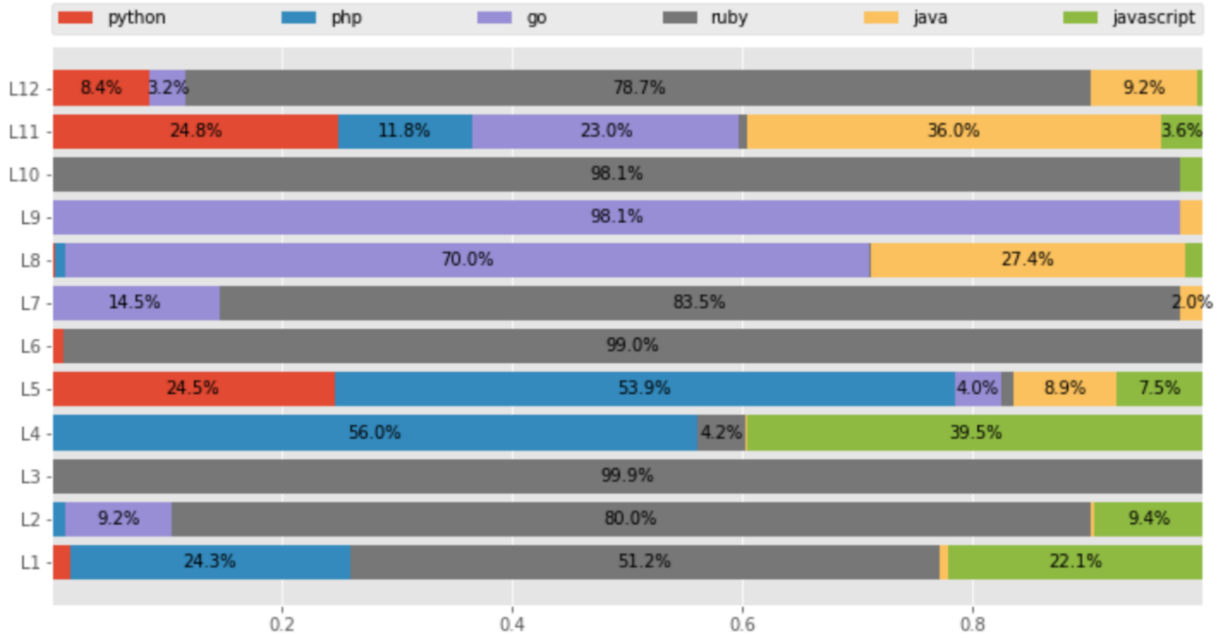

Add & Norm
AdvFusion Adapter
Key
Value
Query
Lang$_1$ Adapter
Lang$_k$ Adapter
Lang$_n$ Adapter
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention


Add & Norm
AdvFusion Adapter
Key
Value
Query
Lang$_1$ Adapter
Lang$_k$ Adapter
Lang$_n$ Adapter
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention

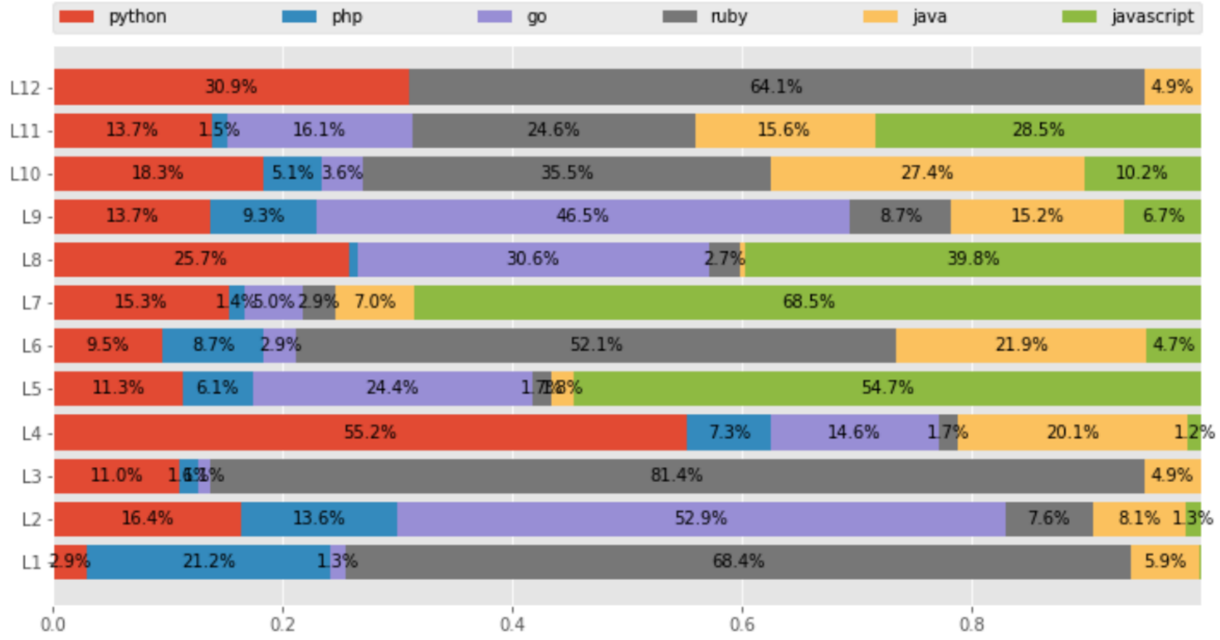
Models	Ruby	JavaScript	Go	Python	Java	PHP
GraphCodeBERT+AdvFusion	16.47	15.89	19.96	18.49	18.97	24.83
GraphCodeBERT+Fusion	15.57	14.49	18.21	17.86	18.31	23.54
GraphCodeBERT+TaskAdapter	14.39	14.53	18.47	17.88	17.29	23.36
CodeBERT+AdvFusion	**16.53**	**16.80**	**24.09**	18.28	19.94	25.20
CodeBERT+Fusion	15.77	16.22	24.01	18.40	19.85	25.17
CodeBERT+TaskAdapter	14.12	15.67	23.21	18.47	18.99	25.55
*polyglot*GraphCodeBERT [5]	14.95	15.79	18.92	18.90	19.91	26.15
*polyglot*CodeBERT [5]	14.75	15.80	18.77	18.71	20.11	**26.23**
CodeT5 [55]	15.69	16.24	19.76	**20.36**	**20.46**	26.09
GraphCodeBERT	12.62	14.79	18.40	18.02	19.22	25.45
CodeBERT	12.16	14.90	18.07	19.06	17.65	25.16
RoBERTa [61]	11.17	11.90	17.72	18.14	16.47	24.02

Language	*polyglot*CodeBERT (# trainable params:110M)	*AdvFusion*CodeBERT (# trainable params:21M)	Improvement
Ruby	14.75	16.53	12%
Javascript	15.80	16.80	6%
Go	18.77	24.09	28%
Python	18.71	18.28	0%
Java	20.11	19.94	0%
PHP	26.23	25.20	0%

Language	*polyglot*CodeBERT Training Time (20000 steps)	*AdvFusion*CodeBERT Training Time (20000 steps)	Time reduction
Ruby	8:06	4:09	-48% ↓
Javascript	8:05	4:22	-45% ↓
Go	8:07	4:50	-40% ↓
Python	8.03	5:09	36% ↓
Java	8:04	4:27	-44% ↓
PHP	8:06	4:47	-41% ↓



Without AdvFusion



With AdvFusion

Adapters are useful for **Multi-modal** knowledge transfer (NL to PL)

Changing adapter architecture **SE-specific adapters**

1) Impose new info to LM
2) Knowledge transfer **among PLs**

LLM-Based Agents

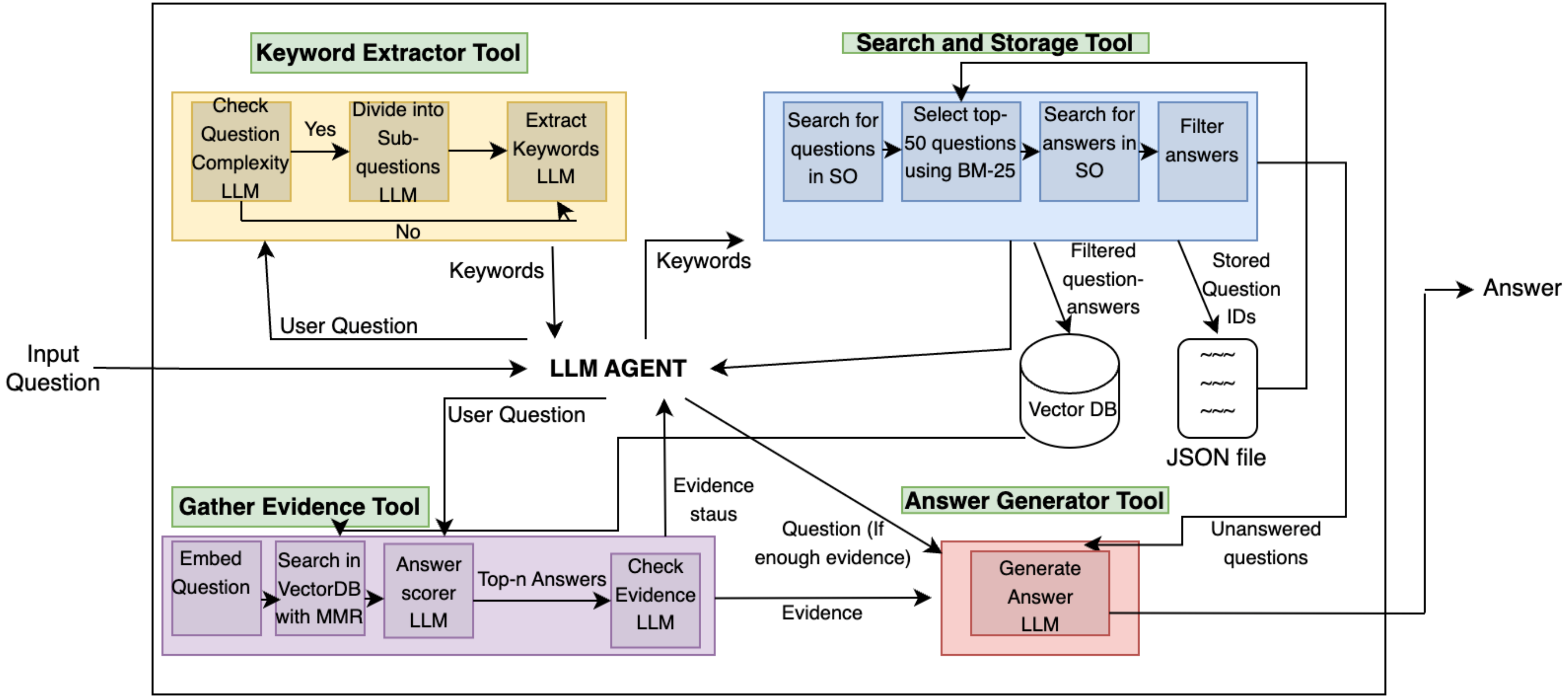
stack overflow

VS

ChatGPT

UBC

AI-Agent


Keyword Extractor Tool
Check Question Complexity LLM
Yes
Divide into Sub-questions LLM
Extract Keywords LLM
No
Search and Storage Tool
Search for questions in SO
Select top-50 questions using BM-25
Search for answers in SO
Filter answers
Keywords
Keywords
User Question
Filtered question-answers
Stored Question IDs
Input Question
LLM AGENT
Vector DB
~~~
~~~
~~~
JSON file
Answer
User Question
Evidence staus
Gather Evidence Tool
Answer Generator Tool
Question (If enough evidence)
Unanswered questions
Embed Question
Search in VectorDB with MMR
Answer scorer LLM
Top-n Answers
Check Evidence LLM
Evidence
Generate Answer LLM
~~~

ChatGPT 3.5

D **You**

what is Llama2 ?

ChatGPT

As of my last update in January 2022, there isn't a widely recognized concept or entity called "Llama2." It's possible that "Llama2" refers to something specific that has emerged after my last update, such as a product, technology, project, or term within a particular context. Without further context, it's challenging to provide a precise answer. If "Llama2" is a new term or concept, I would need more information to provide an accurate explanation.

UBC

AccurAnswer

What is Llama2?

Llama2 is a collection of pretrained and fine-tuned large language models (LLMs) ranging in scale from 7 billion to 70 billion parameters. These models, called Llama2-Chat, are optimized for dialogue use cases. They have been shown to outperform open-source chat models on various benchmarks and have been evaluated for helpfulness and safety. Llama2 is part of the LLaMA project, which aims to develop open and efficient foundation language models.

Evaluation metrics and Results

Ragas Score

- **Faithfulness (F)**
- **Answer Relevance (AR)**
- **Context Relevance (CRel)**
- **Context Recall (Crec)**

	F	AR	CRel	CRec	RAGAS
512 wo/ rerank	0.7182	0.9337	0.2589	0.7190	0.5184
512 w/ rerank	0.7485	0.9619	0.3528	0.7690	0.6144
1024 wo/ rerank	0.7603	0.9270	0.2589	0.6857	0.5185
1024 w/ rerank	0.8688	0.9648	0.3511	0.8006	0.6364

Table 1: RAGAS scores. The numbers 512 and 1024 refer to the chunk sizes. For each chunk size, two numbers are reported, one without (wo/) reranking and the other one using a reranking technique (w/). In all cases, $K = 1$.

Tonic Metrics

- **Answer Similarity (AS)**
- **Retrieval Precision (RP)**
- **Augmentation Precision(AP)**
- **Augmentation Accuracy (AA)**
- **Answer Consistency (AC)**

	AS	RP	AP	AA	AC	Overall
512 wo/ rerank	3.48	0.70	0.60	0.63	0.61	0.65
512 w/ rerank	4.2	1.0	0.77	0.79	0.77	0.83
1024 wo/ rerank	3.48	0.700	0.60	0.63	0.61	0.65
1024 w/ rerank	4.1	0.97	0.93	0.93	0.92	0.92

Table 2: Tonic Metric Scores. The numbers 512 and 1024 refer to the chunk sizes. For each chunk size, two numbers are reported, one without (wo/) reranking and the other one using a reranking technique (w/). In all cases, $K = 1$.

Knowledge Transfer for Software Engineering

- From PL-LLMs with SE-specific adapters
- From QA platforms using RAG LLM-based agents
- Change
  - our point of view,
  - architecture, or
  - use the current knowledge sources

Thank You ☺