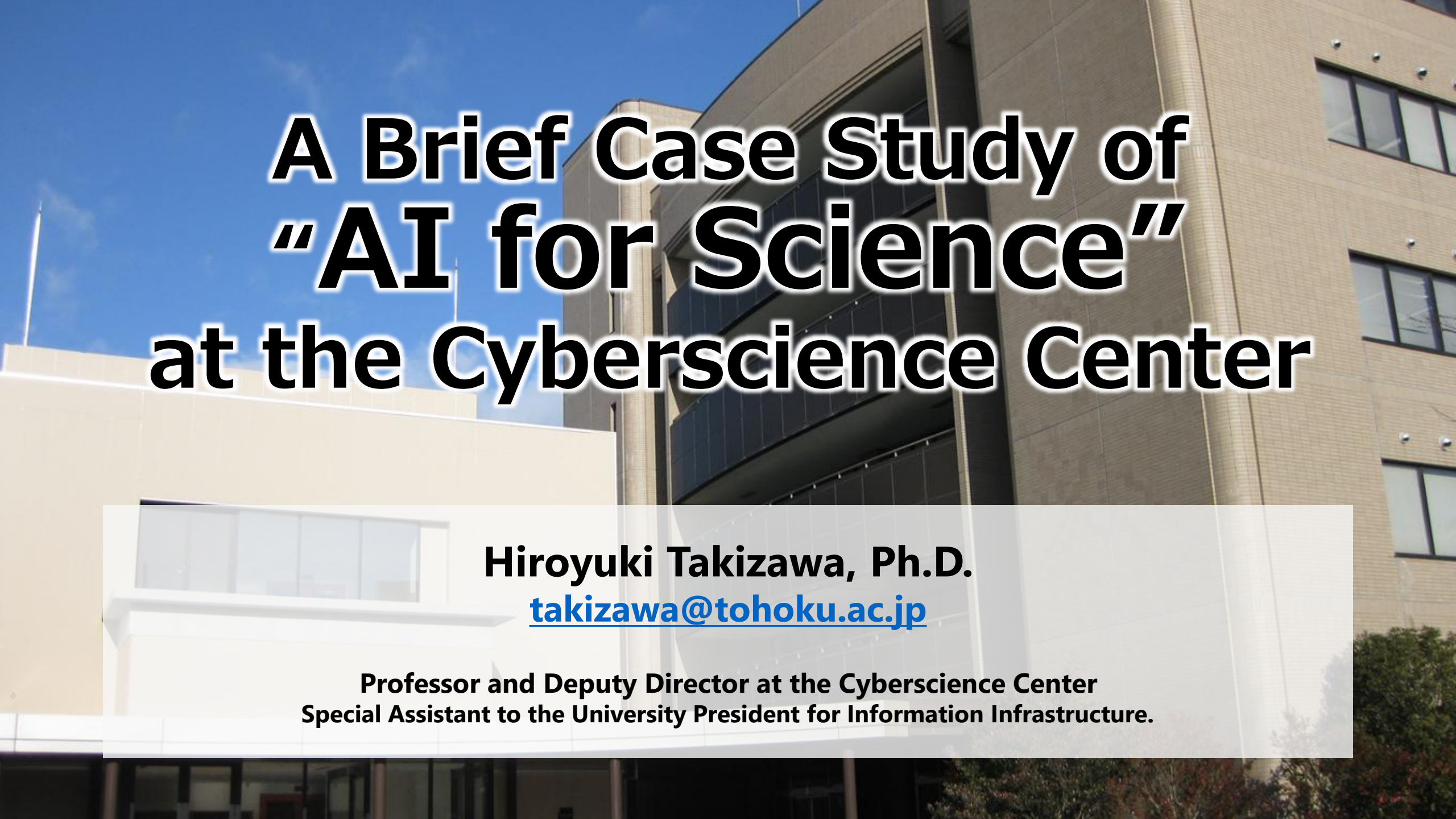


A Brief Case Study of "AI for Science" at the Cyberscience Center

Hiroyuki Takizawa, Ph.D.

takizawa@tohoku.ac.jp

Professor and Deputy Director at the Cyberscience Center

Special Assistant to the University President for Information Infrastructure.

Development and deployment of the next flagship system succeeding “Fugaku”

Effective performance up to 100× power efficiency about 20×

New Flagship System

Expected to start operation by around 2030

Supercomputer “Fugaku”

- **System**
  - Development : **RIKEN**
  - **160k high performance CPUs**
  - **High effectiveness and versatility** through co-development with application software
- **Performance**
  - Theoretical peak performance: 537PFlops, System power consumption: 30MW
  - **No.1 in 4 global rankings for 4 consecutive terms** (Jun 2020 – Nov 2021), **11 consecutive No.1 titles** in one category (Jun 2020 – Jun 2025 latest)
- **Operation and Utilization Status**
  - **Consistently achieving extremely high availability (over 95%),** Industrial use on the rise – within 3 years of shared use, more corporate users than the total accumulated over 8 years with “K computer”
- **Use Case**

New Flagship System

- **System Overview and Performance Indicators**
  - Development : **RIKEN**
  - Leveraging simulations on the current Fugaku, delivering **over 5–10 times** the effective performance
  - Performance required for AI training and inference → **World-class computing environment** (effective performance 50+ EFLOPS)
  - Enhanced power efficiency enabling the above computing environment
  - CPUs complemented with **accelerators** (e.g., GPUs)
- **Development & Expansion Initiatives**
  - **Minimize transition gaps**, maintaining user environment
  - Flexible replacement/expansion for **continuous evolution**
  - Ongoing R&D of technologies for future demand

- Progress Status
  - Development started in January 2025
  - Fujitsu Ltd. is selected as CPU and system integration vendor
  - NVIDIA is selected as accelerator vendor

Kurihara at WSSP40

tom's HARDWARE

US Edition RSS Search

Best Picks | Raspberry Pi | CPUs | GPUs | 3D Printers | More ▾ | Forums

Japan to begin developing ZetaFLOPS-scale supercomputer in 2025

News By Anton Shilov published August 27, 2024

Well, 'AI' ZetaFLOPS.

(Image credit: Lenovo)

Japan's Ministry of Education, Culture, Sports, Science and Technology (MEXT) has announced [PDF] plans to build a successor to the country's Fugaku supercomputer, which was once the world's most powerful HPC machine. The ministry wants RIKEN and Fujitsu to start developing the supercomputer next year, reports Nikkei.

文部科学省

17

Kurihara at WSSP40

HPCI framework in Japan

HPCI Core Computing Resources (14 Institutions)

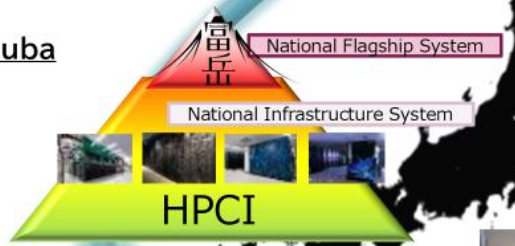
JCAHPC
Miyabi-G/ Miyabi-C Ⓖ

Nagoya University
Flow Type-I/II Ⓖ

Institute of Science Tokyo
TSUBAME4.0 Ⓖ

Hokkaido University Ⓖ
Grand Chariot 2

University of Tsukuba
Pegasus Ⓖ



JHPCN

Kyushu University
Genkai node group A/B Ⓖ

Tohoku University
AOBA-A/B/S Ⓥ

- IMR (MASAMUNE-Ⅱ) Ⓖ
- IFS (AFI-NITYⅡ) Ⓖ
- ToMMo(ToMMo Supercomputer) Ⓖ

The University of Tokyo
Wisteria/BDEC-01 (Aquarius) Ⓖ
Wisteria/BDEC-01 (Odyssey)

Ⓖ
- ISSP(Ohtaka, Kugui)
- IMSUT(Shirokane7)

Ⓖ

Kyoto University
Camphor3
(NVIDIA A100 GPU installed in SystemG Gardenia, but only CPU node publicly utilized)

- YITP Ⓖ (Yukawa-21)
- ICR Ⓖ (HPE Superdome Flex, Apollo2000, DL380)
- OUBIC Ⓖ

RIKEN
FUGAKU
HOKUSAI BigWaterfall2

AIST
ABCI 3.0 Ⓖ

JAMSTEC
Earth Simulator (ES4) Ⓥ Ⓖ

ISM
A

The University of Osaka
SQUID Ⓥ Ⓖ

- QST・NIFS (Plasma Simulator) Ⓖ

- NIMS(Numerical Materials Simulator)
- NIED(Cray CS500, CS-Storm Ⓖ)
- RIKEN(Supercomputer System for "AI for Science" Development Ⓖ, RAIDEN Ⓖ)
- JAXA(TOKI-SORA)
- JAEA・QST(HPE SGI8600) Ⓖ

National Research and Development Agency

- NIPR(Polar Science Computing System)
- NIG(TeslaV100 Ⓖ, NVIDIA Blackwell B200 Ⓖ)
- ISM(I)
- NAOJ(ATERUI-Ⅲ)
- IMS(High-Performance Molecular Simulator) Ⓖ

Inter-University Research Institute Corporation

Affiliated Institution in National University

Independent data platform

- mdx Ⓖ
- mdxⅡ Ⓖ

Ⓖ :GPU installed

4

Kurihara at WSSP40

AI for Science に不可欠な計算基盤の環境整備

令和7年度補正予算額　76億円

文部科学省

事業目的・概要

科学基盤モデルの開発・利用等の研究活動におけるAI利活用（AI for Science）には、GPUを搭載した膨大な計算資源を有する計算基盤が必要不可欠である。全国14機関が有する計算資源の共用の枠組みである革新的ハイパフォーマンス・コンピューティング・インフラ（HPCI）の利用状況は既に逼迫しており、**AI for Scienceの推進に向けた計算資源の戦略的な増強及び利用環境の整備**が喫緊の課題となっている。

事業内容

①「AI for Scienceによる科学研究革新プログラム」等の取組に必要となる計算資源の確保に向けて、**共用計算資源等の増強に向けた取組を支援**する。

②HPCIの共用計算資源の利用促進を図るために、**現行の利用申請システムの抜本的改修**を行う。

HPCI共用計算資源

事業スキーム

①

国 ⇒（補助）HPCI加盟機関等（大学・国立研究開発法人）

【支援内容】
件数：２～３件程度
単価：最大50億円程度
交付先：HPCI加盟機関（大学、国立研究開発法人）等
を想定

※1　１件当たりおおむね500GPU規模の計算資源を、既存のセンター設備も活用しつつ、効果的・効率的に整備することを想定

※2　最新世代GPUを搭載し、１件当たり 約４～５EFLOPS級（AI性能換算）を想定

②

国 ⇒（委託）一般財団法人 高度情報科学技術研究機構

【システム改修のポイント】
・ユーザインターフェースの利便性向上
・スマートフォンやタブレット等による課題申請の対応
・運用側で機能を追加可能にするなどシステムの柔軟化
・申請者ごとの課題管理の一元化

（担当：研究振興局参事官（情報担当）付）

https://www.mext.go.jp/b_menu/boshu/detail/mext_00503.html

Semantic equivalence verification of codes

This work has been presented at LLM4HPC Asia 2026.
Yuta Tanizawa et al.: Semantic Equivalence Verification of HPC Codes Using LLMs

Motivation

- **Why equivalence verification matters**
  - Refactoring
  - Language migration (Fortran → Python/C++)
  - Code optimization
  - LLM-generated codes

- **HPC codes are large and complex**
  - Legacy Fortran
  - Scientific simulations
  - Long codebases

Background

- 4 types of code equivalence
  - Type1: lexical
    - Identical except for whitespace or comments
  - Type2: syntactic
    - Different identifiers but identical structure
  - Type3: near miss
    - Partially different but highly similar
  - Type4: semantic
    - Different structures but functionally equivalent

```
int arith_sum(int n) {
    int total = 0;
    for (int i = 1; i <= n; i++) {
        total += i;
    }
    return total;
}
```

Different structures and same functionality

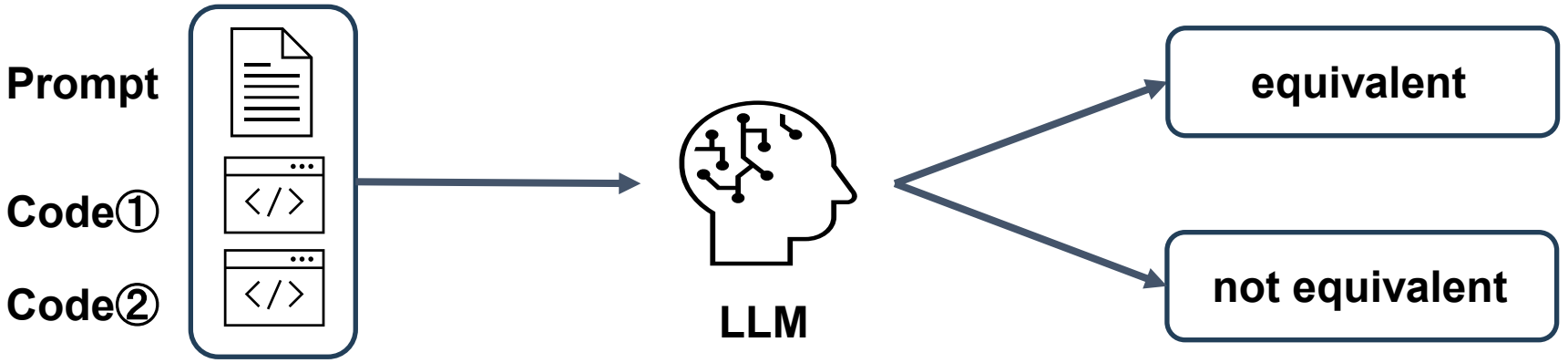
```
int arith_sum(int n) {
    return n * (n + 1) / 2;
}
```

Challenges and Objective

- **Semantic equivalence verification is difficult with conventional methods:**
  - Syntax-based analysis fails for structurally different codes
  - Test-based methods lack completeness
  - Symbolic execution suffers from path explosion

Use of LLMs for Linking **Code Structure with Its Semantic Meaning**

To verify semantic equivalence of source code using LLMs



Related Work

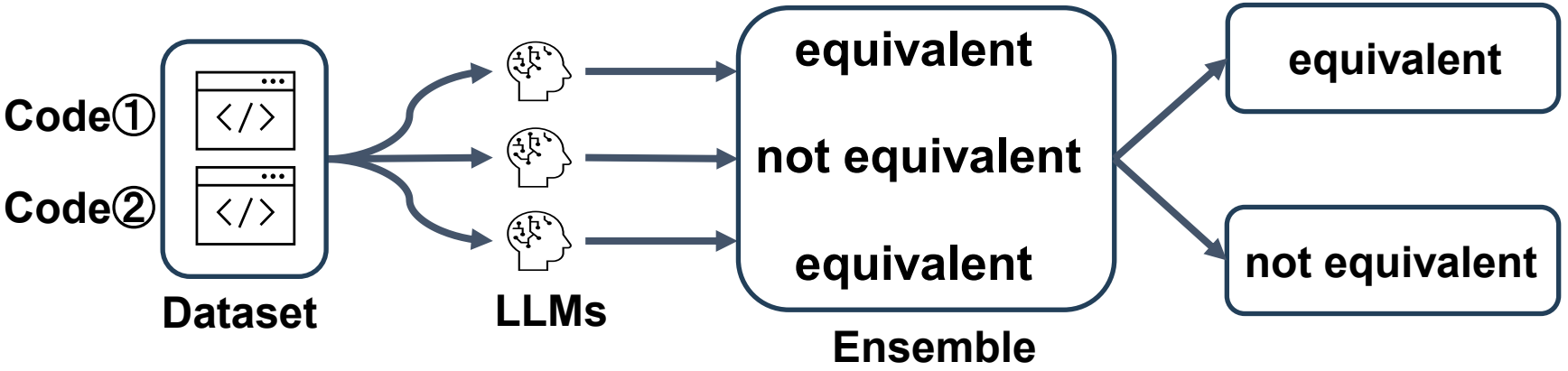
- **Prompt-based semantic equivalence verification using LLMs** (Zhang et al., 2024)
  - A performance comparison between GPT-3.5 and GPT-4
  - Benchmarks:
    - **BigCloneBench** (Svajlenko et al., 2015)
    - **GPTCloneBench** (Alam et al., 2023)

Indicating the Potential of LLMs in Semantic Equivalence Verification

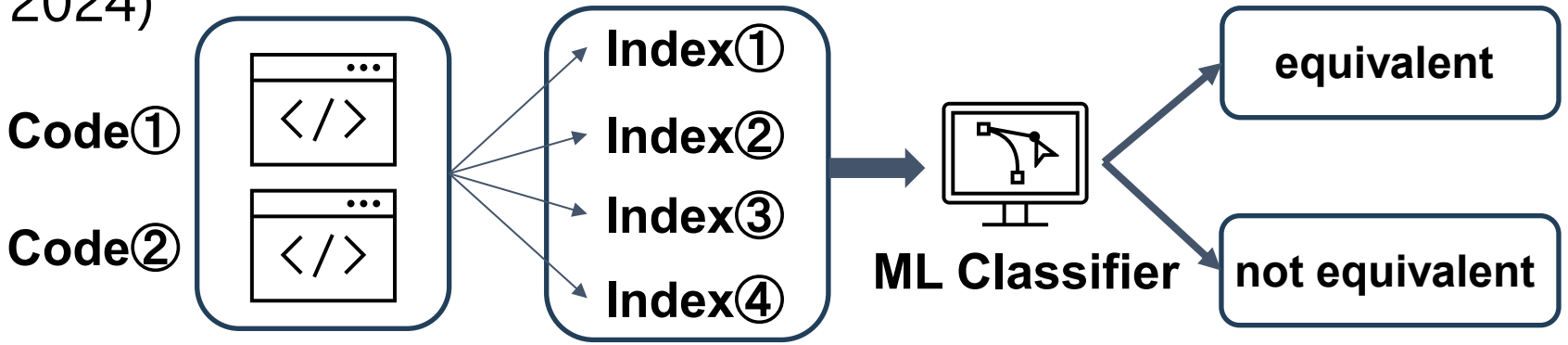
- **Limitations**
  - LLMs are treated as **black boxes**
  - Existing benchmarks focus on **limited languages** (Java, Python)
  - Limited applicability to **HPC codes such as Fortran**

Related Work (Cont'd)

- **Ensemble of multiple prompt-based LLMs using majority voting** (Ahmed et al., 2023)



- **Ensemble of multiple similarity metrics with ML classifiers** (Feng et al., 2024)



Lack of case studies and insufficient coverage

Open Issues in Prior Studies

1. **Semantic equivalence verification for HPC codes (e.g. Fortran) has not been sufficiently evaluated.**

2. **LLMs are used only at the output level**
   - No access to confidence or reasoning
   - Internal representations are not utilized

3. **Ensemble strategies are underexplored**
   - Choice of LLMs
   - Choice of hidden layers
   - Choice of ML classifiers

Contributions of This Work

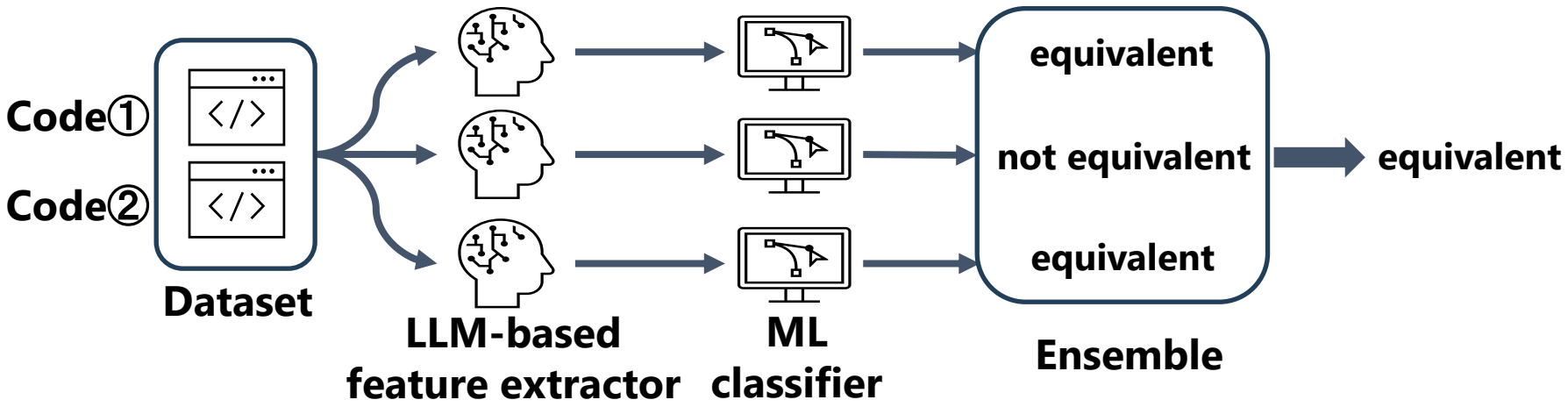
1. **First evaluation of semantic equivalence verification for HPC codes**

2. **Feature extraction from LLM hidden layers**

3. **Systematic exploration of LLM–layer–ML combinations**

Overview of the Proposed Method

- **Key Idea**
  - Feature extraction from LLM hidden layers for ML classifiers
  - Majority-voting ensemble
- **Workflow**
  - Construct a custom dataset
  - Extract hidden-layer representations from LLMs
  - Verify equivalence using pretrained ML classifiers
  - Explore optimal model combinations (n = 3)



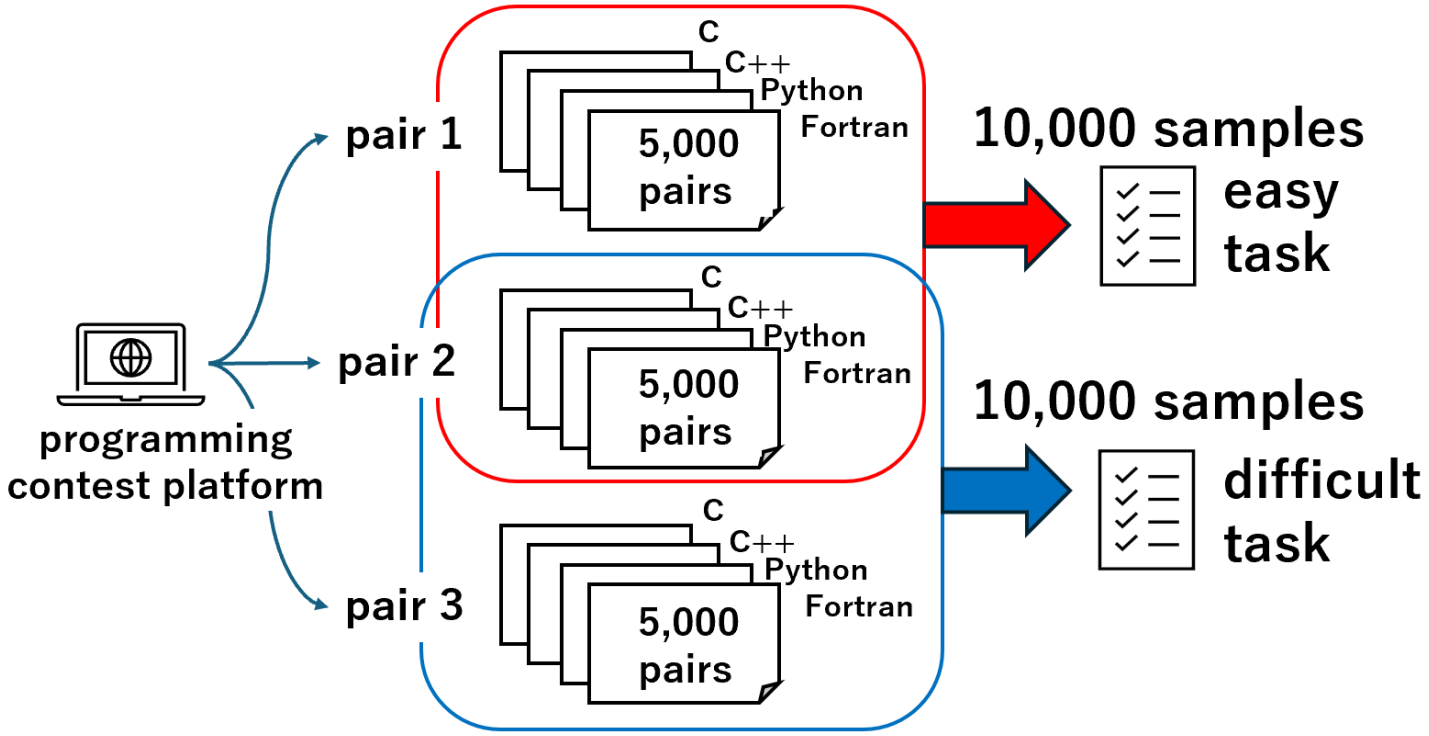
Dataset

- **Source code pairs collected from AtCoder submissions**
  - 15000 code pairs
  - 4 languages
    - Fortran
    - C
    - C++
    - Python

- **AtCoder is a programming contest**
  - Submitted codes could be incorrect for a given problem.
  - Codes for **different problems** → **obviously different** = easy to verify
  - Codes for **the same problem** → **likely similar** = difficult to verify

Dataset Construction

- **Code Pair Types**
  - Pair Type 1: Semantically non-equivalent (different problems)
  - Pair Type 2: Semantically equivalent (same problem, both are correct)
  - Pair Type 3: Semantically non-equivalent (same problem, one is incorrect)



A **custom dataset** was constructed due to the lack of Fortran benchmarks.

※ This talk focuses only on **difficult tasks** to clearly observe performance differences

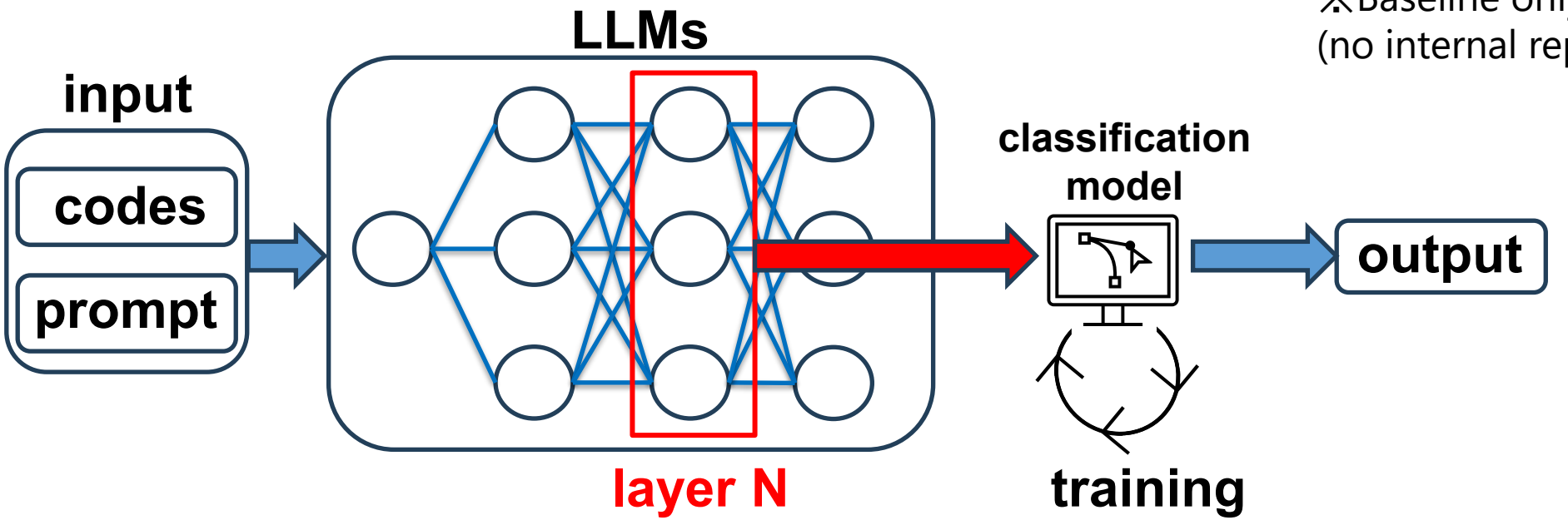
Feature Extraction Using LLMs

- Input: Code pairs + prompt
- Output: Semantic vectors from hidden layers

Using **hidden layers of LLMs** for equivalence verification is novel

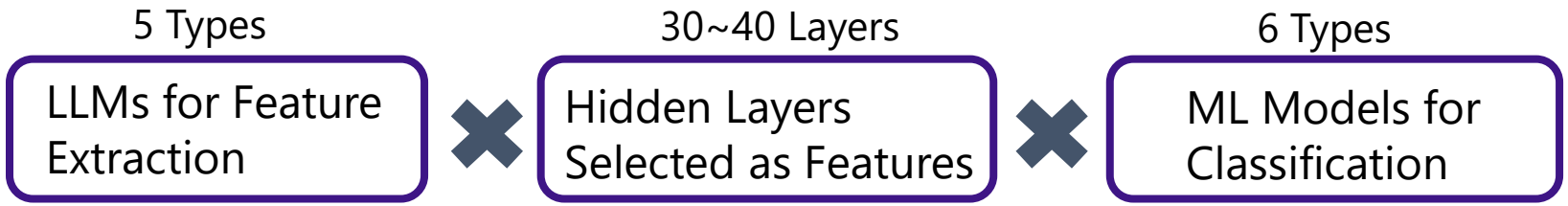
- **LLMs Used**
  - DeepSeek-Coder
  - OpenCoder
  - Llama
  - Phi-4
  - Qwen2.5-Coder
  - (ChatGPT-4o)

  ※Baseline only
  (no internal representations)



ML Classifiers and Ensemble

- **ML Classifiers**
  - K-Nearest Neighbors (KNN)
  - LightGBM (LGBM)
  - Multi-Layer Perceptron (MLP)
  - Random Forest (RF)
  - Support Vector Machine (SVM)
  - XGBoost (XGB)
- **Ensemble Strategy**
  - The final binary outputs from the ML models are evaluated using a majority-voting ensemble



For 30 LLM–ML combinations, the top 3 hidden layers are selected for each, yielding 90 candidates ($_{90}C_3$ = 117,480)

Such systematic evaluations of **LLM–layer–ML combinations** remain unexplored.

Evaluation Conditions

- **Evaluation Overview**
  - Comparison with conventional methods (Fortran only)
    - Proposed Method:
      - Majority-voting ensemble of ML classifiers with LLM features
    - Baseline Method
      - Majority-voting ensemble of prompt-based LLM's outputs
  - Ablation Study
    - Prompt-based LLM
      (evaluated on Fortran, Python, and C/C++)
    - ML classifiers with LLM features
      (evaluated on Fortran only)
- **Evaluation Environment**
  - CPU: AMD EPYC 7352
  - GPU: NVIDIA A100 80GB
  - Memory: 480 GiB of DDR SDRAM

Overall Comparison with Baselines

- **Comparison Results**
  - The F1 score improves by 0.265
  - Intermediate layers are effective for feature extraction
  - MLP achieves the best performance among ML classifiers

Conventional Method

model combination	Precision	Recall	Accuracy	F1 Score
ChatGPT + llama + OpenCoder	0.585	0.925	0.590	**0.717**
llama + OpenCoder + phi-4	0.560	0.987	0.566	0.715
DeepSeek-Coder + OpenCoder + phi-4	0.607	0.859	0.628	0.711
ChatGPT + OpenCoder + phi-4	0.643	0.789	0.670	0.709
OpenCoder + phi-4 + Qwen2.5-Coder	0.607	0.844	0.632	0.706

+0.265

Proposed Method

model combination	Precision	Recall	Accuracy	F1 Score
MLP_DeepSeek-Coder_L16+MLP_OpenCoder_L18+MLP_phi-4_L11	0.972	0.972	0.978	**0.972**
KNN_DeepSeek-Coder_L13+MLP_OpenCoder_L18+MLP_phi-4_L22	0.972	0.972	0.978	**0.972**
KNN_DeepSeek-Coder_L11+MLP_OpenCoder_L18+MLP_phi-4_L22	0.972	0.972	0.978	**0.972**
MLP_Llama_L13+MLP_OpenCoder_L18+MLP_phi-4_L22	0.984	0.953	0.971	0.968
MLP_Llama_L13+KNN_OpenCoder_L19+MLP_phi-4_L28	0.968	0.953	0.963	0.961

Use of each LLM with Prompts

- **Evaluation Results**
  - No significant performance gap between Fortran and other languages
  - Fortran F1 score ranges from 0.504 to 0.705
  - Recall and precision exhibit model-dependent biases
    - Llama shows higher recall
    - ChatGPT and Qwen2.5-Coder achieve higher precision

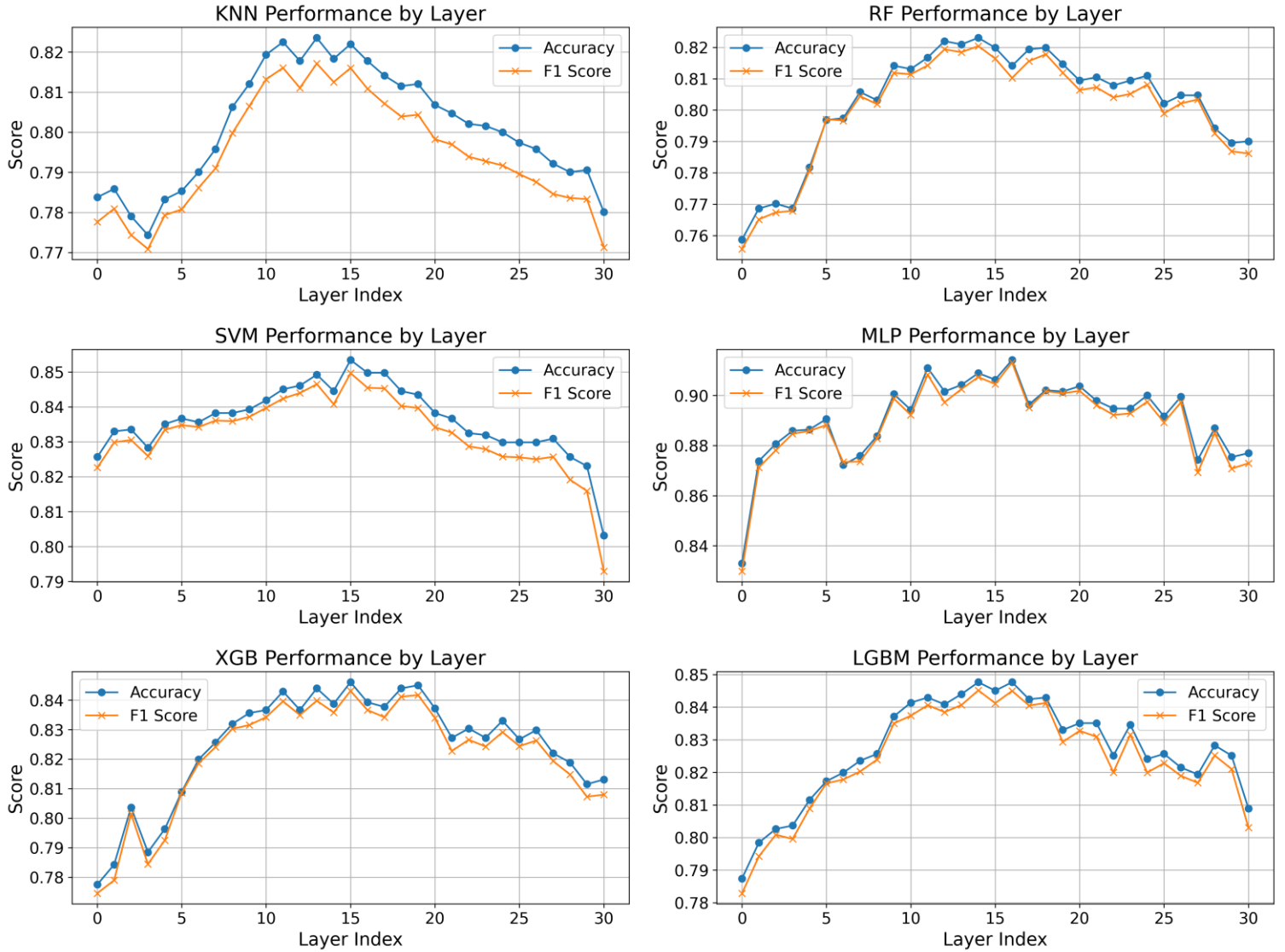
model	lunguage	Precision	Recall	Accuracy	F1 Score
ChatGPT	Fortran	0.630	0.420	0.587	0.504
	Python	0.780	0.533	0.691	0.633
	C++	0.608	0.456	0.581	0.521
	C	0.638	0.407	0.588	0.497
DeepSeek-Coder	Fortran	0.545	0.494	0.538	0.518
	Python	0.692	0.739	0.704	0.715
	C++	0.535	0.733	0.547	0.618
	C	0.561	0.700	0.559	0.623
OpenCoder	Fortran	0.574	0.915	0.576	0.705
	Python	0.539	0.771	0.539	0.634
	C++	0.533	0.908	0.544	0.671
	C	0.571	0.790	0.563	0.663
Llama3	Fortran	0.505	0.980	0.510	0.667
	Python	0.509	0.975	0.518	0.669
	C++	0.499	0.973	0.502	0.660
	C	0.500	0.967	0.501	0.659
Phi-4	Fortran	0.578	0.806	0.611	0.673
	Python	0.662	0.811	0.699	0.729
	C++	0.546	0.839	0.577	0.662
	C	0.569	0.748	0.592	0.646
Qwen2.5-Coder	Fortran	0.612	0.584	0.738	0.598
	Python	0.661	0.649	0.772	0.655
	C++	0.640	0.605	0.755	0.622
	C	0.652	0.521	0.748	0.580

LLM as a Feature Extractor

DeepSeek-Coder

Y-axis:
- **Accuracy**
- **F1 Score**

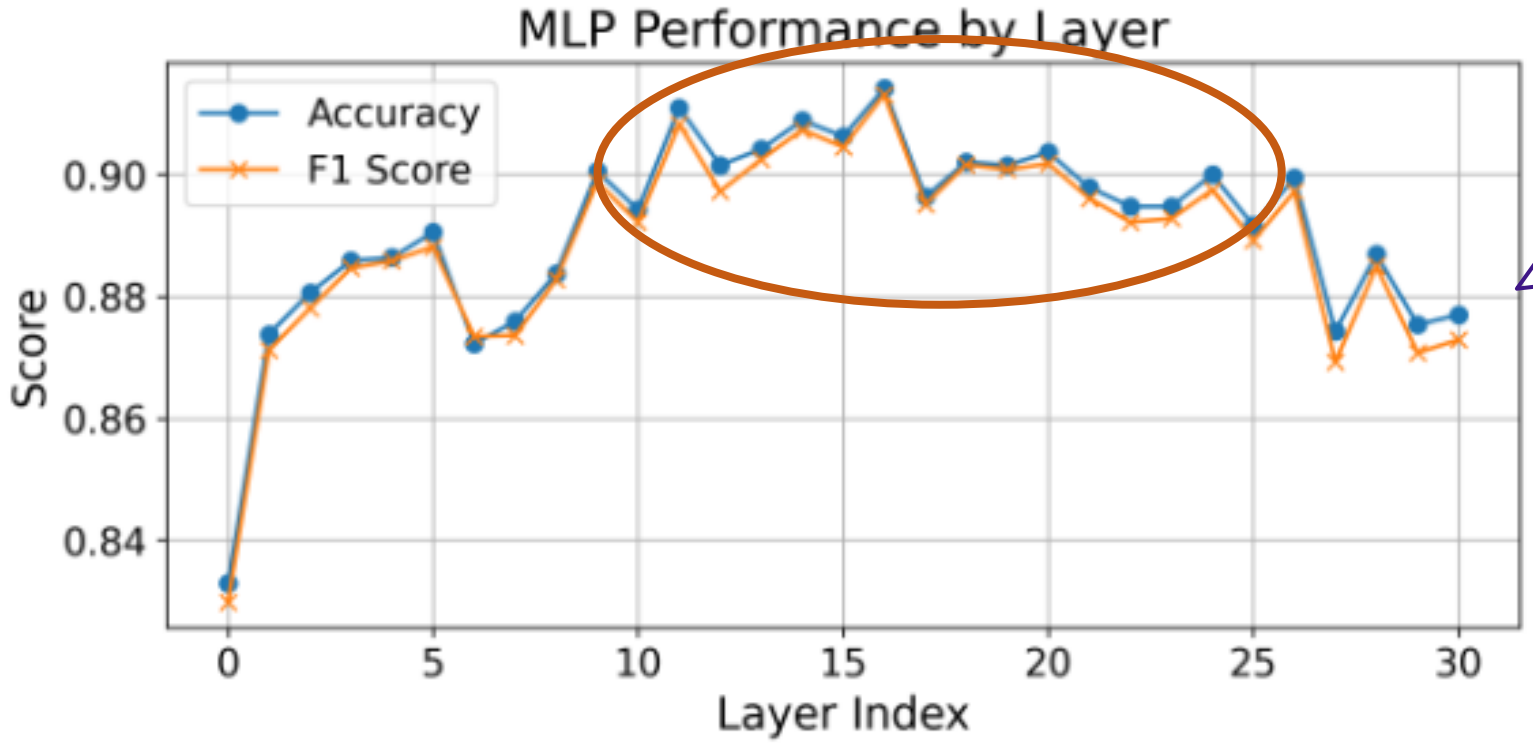
X-axis:
- **Layer Index**



Internal representations at intermediate layers are useful for equivalence verification

- **Evaluation Results**
  - Best F1 score: **0.913**
  - Intermediate layers show relatively high performance
  - MLP consistently achieves the best performance across LLM–ML combinations

DeepSeek-Coder × MLP



Discussion

Shallow layers capture lexical information. Deeper layers are optimized for generation tasks.

Similar trends are observed with other LLMs

Summary

- **Background**
  - LLM-based methods are promising for code semantic equivalence verification
- **Challenges**
  - HPC codes (e.g., Fortran) remain underexplored
  - Existing methods rely on surface-level, black-box LLM outputs
  - Ensemble combinations are insufficiently studied
- **Objective**
  - To achieve semantic equivalence verification of source code using LLMs
- **Proposed Method**
  - Majority-voting ensemble combining LLM features and ML classifiers
- **Conclusions**
  - Achieves robust performance on HPC datasets
    - Best F1 score: 0.972 (vs. baseline 0.717)
    - Intermediate LLM layers are most effective
    - MLP performs best among evaluated classifiers

Future Work

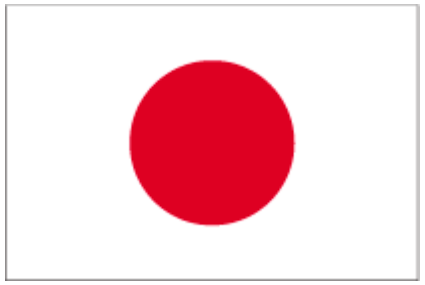
- **Detailed analysis of inference cost**
  - Evaluate the computational cost of feature extraction and the ensemble pipeline
  - Visualize the accuracy–cost trade-off by comparing with baseline methods
- **Evaluation on real-world problems**
  - Due to dataset construction costs, evaluation is limited to programming contest datasets
  - Future work includes validation on real-world HPC codes (MPI/OpenMP, simulations, legacy Fortran)

Structural Differences in AI Investment?

Policy

EU framework

Ecosystem



Infrastructure

HPC Center

Enhancement

Industry

Big Tech

Scaling