

AI for software development

Tong Liu¹, Xuliang Zhu², Yue Sun¹, Shenchun Wang¹,
Zhiwen Tan, Bolun Zhang¹, Ke Li¹

¹IHEP

²SJTU

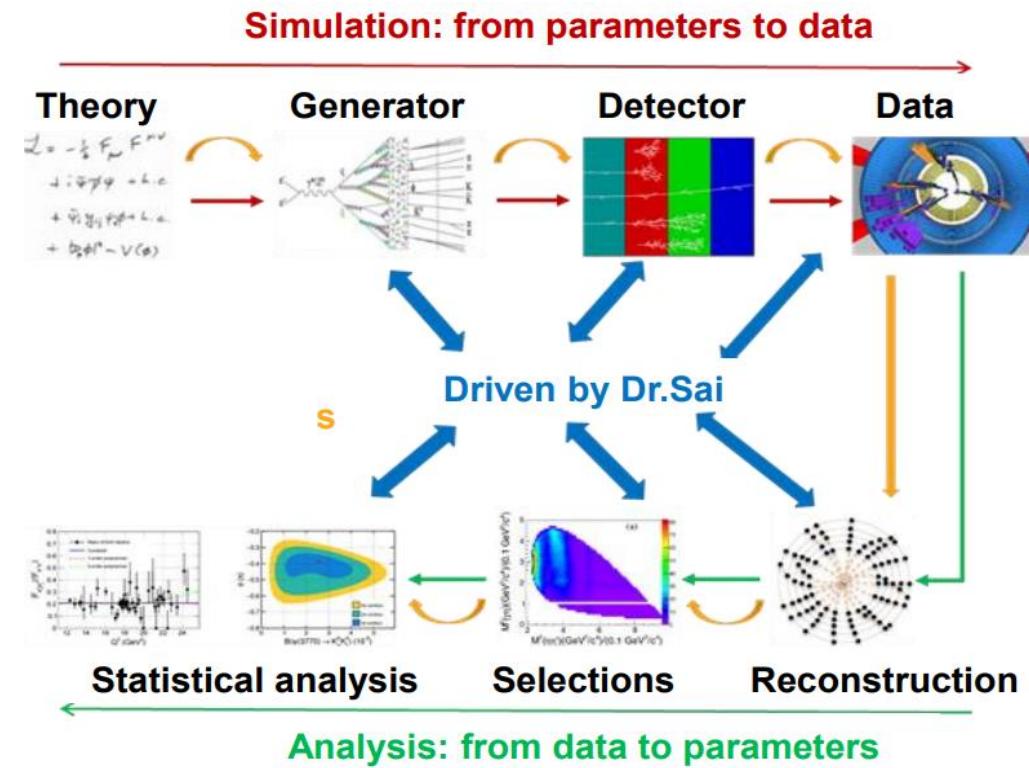
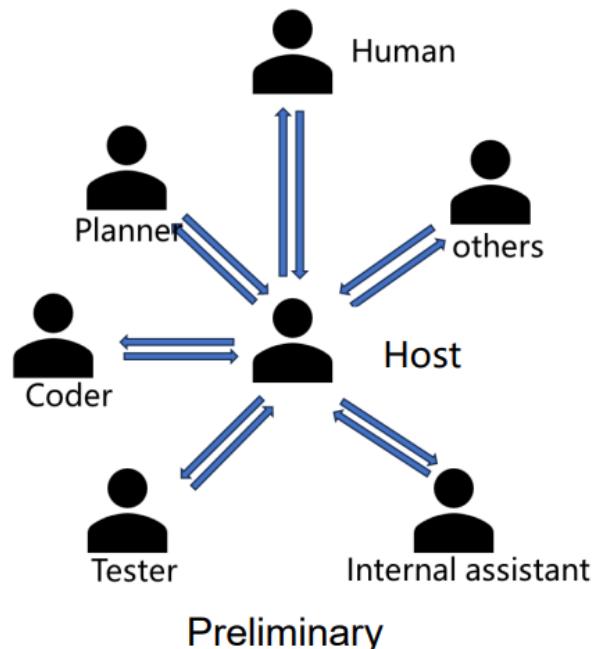
³LNU

Outline

- Introduction
- Data set
- Current status
- Future
- Summary

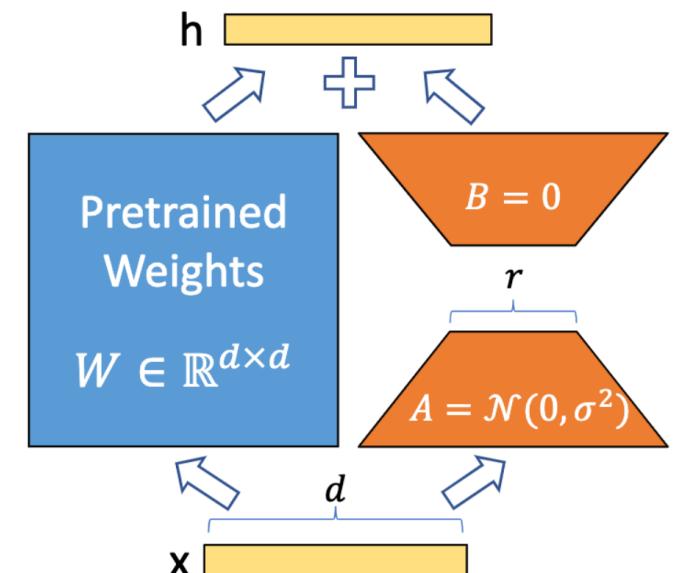
Introduction

- Analysis of HEP is quite complex
 - Lots of scripts, checks, human-computer interactions
- Develop a AI system to speed up and simplify
 - Start from BESIII: Dr. Sai, a multi-agents system
 - Use LLM for text/code generation



Introduction

- As part of the auto-analysis, currently code generation is a big issue
 - ROOT code, BOSS code, ...
 - To enhance code generation capabilities, perform a model fine-tuning
- We use LoRA (Low-Rank Adaptation) technology
 - An efficient technique for model fine-tuning, introduces low-rank matrices to reduce the number of parameters
 - Update the full ($d \times d$) matrix \rightarrow update $A(d \times r)$ and $B(r \times d)$
 - Trainable parameters/full parameters = 0.01-1%
 - Performance close to the full-parameter fine-tuning



$$h = W_0x + \Delta Wx = W_0x + BAx$$

Data set

- First focus on fit codes in CERNROOT frame: \$ROOTSYS/tutorials/fit
 - The package only provide C code here
- Build training samples using deepseek-r1:671b
 - Read code, understand the functionality, remove comments
 - QA for the full code (~40 QA pairs)
 - QA for code blocks (~160 QA pairs)
 - Keep Q start with “使用CERNROOT框架” (“with CERNROOT frame”)
 - (next, generate more QA pairs with LLM)

```
{  
  "file": "line3Dfit.C",  
  "comment": "使用CERNROOT框架，沿着一根三维直线产生散点数据，然后拟合得到这跟直线的参数。通过最小化每个点到直线的距离平方和来确定最佳拟合直线。将原始直线与拟合直线进行比较展示。",  
  "code": "/// \file /// \ingroup tutorial_fit /// \notebook Fit  
ting of a TGraph2D with a 3D straight line///\n/// run this macro by doing  
:\n///\n/// ~~~{.cpp}\n/// root>.x line3Dfit.C+\n/// ~~~\n/// \macro_i  
mage\n/// \macro_output\n/// \macro_code\n///\n/// \author Lorenzo Moneta  
\n\n#include <TMath.h>\n#include <TGraph2D.h>\n#include <TRandom2.h>\n#include <  
QA for the full code
```

```
{  
  "comment": "在CERNROOT框架中，定义三维直线参数方程，通过参数t生成坐标点(x,y,z)",  
  "code": "void line(double t, const double *p, double &x, double &y, double &z) {\n    x = p[0] + p[1]*t;\n    y = p[2] + p[3]*t;\n    z = t;\n}"},  
  {  
    "comment": "在CERNROOT框架中，定义距离平方计算结构体，用于拟合过程中最小化点到直线的距离平方和",  
    "code": "struct SumDistance2 {\n    TGraph2D *fGraph;\n    SumDistance2(TGraph2D *g) : fGraph(g) {}  
    double distance2(double x, double y, double z, const double *p) {\n        XYZVector xp(x,y,z);\n        XYZVector x0(p[0],  
QA for code blocks
```

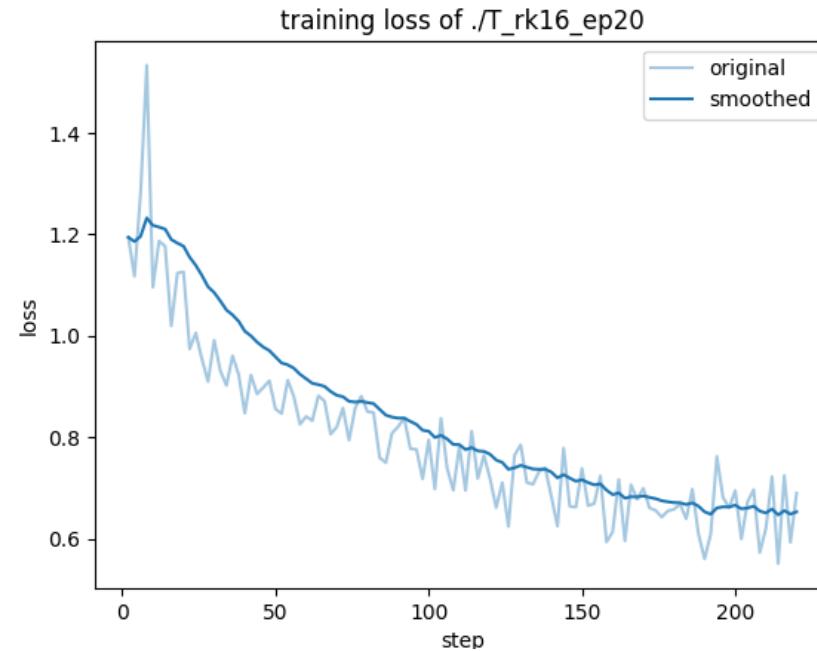
Data set

- Generate 10* code, calculate the number of times there are errors
- Test samples: a few commonly used fit tasks:
 - 1) 2D fit to 2D-Gaussian
 - 2) Linear+Gaussian, multi-range fit in sideband
 - 3) Fit to a TGraphAsymmErrors
 - 4) Fit and print C.L.s
 - 5) Simultaneous fit to sample1 (BG) and sample2 (BG+Sig)
 - 6) Fit to func \otimes Gaussian

Target	Qwen2.5-7B	deepseek-r1:671b
1)	1/10	5/10
2)	2/10	3/10
3)	9/10	9/10
4)	9/10	7/10
5)	5/10	0/10
6)	10/10	6/10

Current status

- Fine-tuning with LLaMA-Factory
 - An open-source toolkit designed to simplify and accelerate the training, fine-tuning, and deployment of LLM
- Considering the current available data samples, we use Lora options:
 - Based on Qwen2.5-7B
 - Rank = 4, 8, 16
 - Epoch = 2, 5



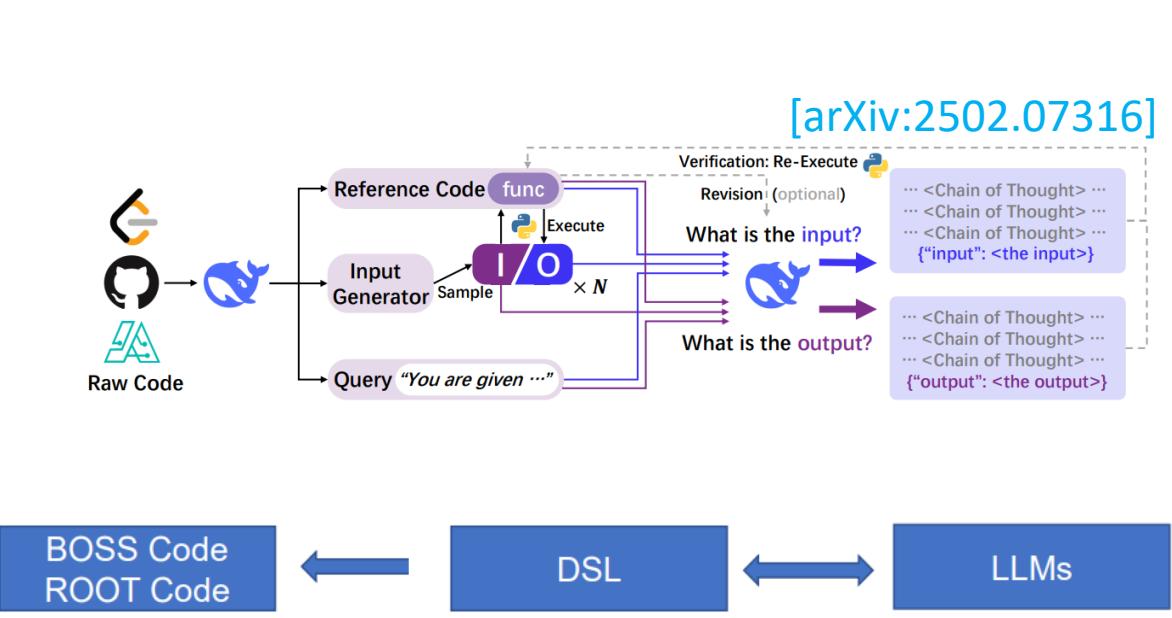
Current status

- Considering the current available data samples, we use Lora options:
 - Based on Qwen2.5-7B
 - Rank = 4, 8, 16
 - Epoch = 2, 5
- The model is improved after fine-tuning
 - Especially for complex tasks

Target	Qwen2.5-7B	deepseek-r1:671b	rk4 ep2	rk4 ep5	rk8 ep2	rk8 ep5	rk16 ep2	rk16 ep5
1)	1/10	5/10	0/10	3/10	1/10	3/10	2/10	3/10
2)	2/10	3/10	4/10	1/10	4/10	2/10	4/10	6/10
3)	9/10	9/10	1/10	1/10	1/10	4/10	0/10	3/10
4)	9/10	7/10	10/10	7/10	6/10	6/10	8/10	8/10
5)	5/10	0/10	1/10	1/10	3/10	1/10	2/10	2/10
6)	10/10	6/10	4/10	8/10	7/10	8/10	3/10	4/10

Future

- CODEI/O, an approach to improve the reasoning abilities of LLMs by training them to predict code inputs and outputs in pure natural language Chain-of-Thought(CoTs)
- Domain Specific Language (DSL4BES)



TASK 3: PID for (anti)-proton. Difficulty: ★★☆☆☆

User input

```
# 10. test pid selection
event_selection.pid(method="probability") do
    prob_cut 0.001
    identify :prp, "from_kaon_and_pion"
    identify :prm, "from_kaon_and_pion"
    nprp ">=1"
    nprm ">=1"
end
```

Ruby IDE output

```
ParticleID* pid = particleID::Instance();
Vint index_ppr; index_ppr.clear();
Vint index_prm; index_prm.clear();
//*****pid (charged tracks)*****
for (int i = 0; i < nchpp; i++) {
    evtRecTrackIterator itTrk = evtRecTrackCol->begin() + iChrgn[i];
    pid->init();
    pid->init();
    pid->setMethodProbability();
    pid->setChiMinCut(4);
    pid->setRecTrack(*itTrk);
    pid->usePdbsys(pid->useDedx() | pid->useOff() | pid->useTof2() | pid->useTof());
    pid->identify(pid->onlyKaon() | pid->onlyPion() | pid->onlyProton());
    pid->calculate();
    if (!pid->isPdbsysValid()) continue;
    if (!itTrk->isMcTrackValid()) continue;
    if (!itTrk->isMcTrackValid()) continue;
    RechMcTrack* mdctrk = (*itTrk)->mdctrack();
    if ((probProton() > probKaon()) && (probProton() > probPion()) && (probProton() > 0.001)){
        RechMcTrack* mdctrk = (*itTrk)->mdctrack();
        RechMcTrack::setPidType(RechMcTrack::probProton());
        index_ppr.push_back(iChrgn[i]);
    }
}
int nprr = index_ppr.size();
int nprm = index_prm.size();
if ((nprr >i)) { return scp }
if ((npmp >i)) { return scc }
```

Summary

- Capabilities on code generation improves after fine-tuning
 - Optimize hyperparameters
 - There is limited tasks available, overfitting?
- We are working on generating more training samples
- Add other tasks like RooFit, BOSS code