AIOps Challenge

2023 CCF国际AIOps挑战赛决赛
暨“大模型时代的AIOps”研讨会

UniLog: Automatic Logging via LLM and In-Context Learning

贺品嘉 香港中文大学（深圳）

主办单位：中国计算机学会（CCF）、清华大学、中国建设银行股份有限公司、南开大学
承办单位：中国计算机学会互联网专委会、清华大学计算机科学与技术系、中国建设银行股份有限公司运营数据中心、南开大学软件学院、北京必示科技有限公司
赞助单位：华为技术有限公司、国网宁夏电力有限公司电力科学研究院、软通动力信息技术（集团）股份有限公司

AIOps Challenge

2023 CCF国际AIOps挑战赛决赛
暨“大模型时代的AIOps”研讨会

目录
CONTENTS

演讲单位logo

第一章节

背景概述

背景概述

AIOps Challenge

日志记录

即，在业务代码中插入合适的日志语句。

日志记录的目标如下：
1. 保证故障信息或关键事件能够被记录在日志中
2. 保证记录的日志能够用于甄别不同的事件或故障
3. 标准化日志规范，减少冗余日志输出造成的开销

```
1| public void processDisconnect(Channel client) {
2|     String clientID = NettyAttrManager.getAttrClientId(client);
3|     cleanWillMessage(clientID);
4|     client.flush();
5|     client.close().addListener(CLOSE);
6|}
```

细分目标#1：在合适的位置处插入日志语句：Line#4

细分目标#2：设置合适的日志等级：LOG.Info()

细分目标#3：记录合适的日志消息："Disconnect successful, clientID: {}"

```
1| public void processDisconnect(Channel client) {
2|     String clientID = NettyAttrManager.getAttrClientId(client);
3|     cleanWillMessage(clientID);
4|     LOG.info("Disconnect successful, clientID: {}", clientID);
5|     client.flush();
6|     client.close().addListener(CLOSE);
7|}
```

背景概述

AIOps Challenge

日志记录的挑战

1. 难以端到端地完成日志记录任务
2. 难以细粒度完成日志记录
3. 受限于较强的假设

```
1| public void processDisconnect(Channel client) {
2|     String clientID = NettyAttrManager.getAttrClientId(client);
3|     cleanWillMessage(clientID);
4|     client.flush();
5|     client.close().addListener(CLOSE);
6|}
```

细分目标#1: 在合适的位置处插入日志语句: Line#4

细分目标#2: 设置合适的日志等级: LOG.Info()

细分目标#3: 记录合适的日志消息: "Disconnect successful, clientID: {}"

```
1| public void processDisconnect(Channel client) {
2|     String clientID = NettyAttrManager.getAttrClientId(client);
3|     cleanWillMessage(clientID);
4|     LOG.info("Disconnect successful, clientID: {}", clientID);
5|     client.flush();
6|     client.close().addListener(CLOSE);
7|}
```

第二章节

方法实现

方法实现

```
<line0> public void processDisconnect(Channel client) {
<line1>     String clientID = NettyAttrManager.getAttrClientId(client);
<line2>     cleanWillMessage(clientID);
<line3>     client.flush();
<line4>     client.close().addListener(CLOSE);
<line5> }
```

传统方法

子任务#1：在合适的位置处插入日志语句：Line#4
子任务#2：设置合适的日志等级：LOG.Info()
子任务#3：记录合适的日志消息："Disconnect successful, clientID: {}"

UniLog

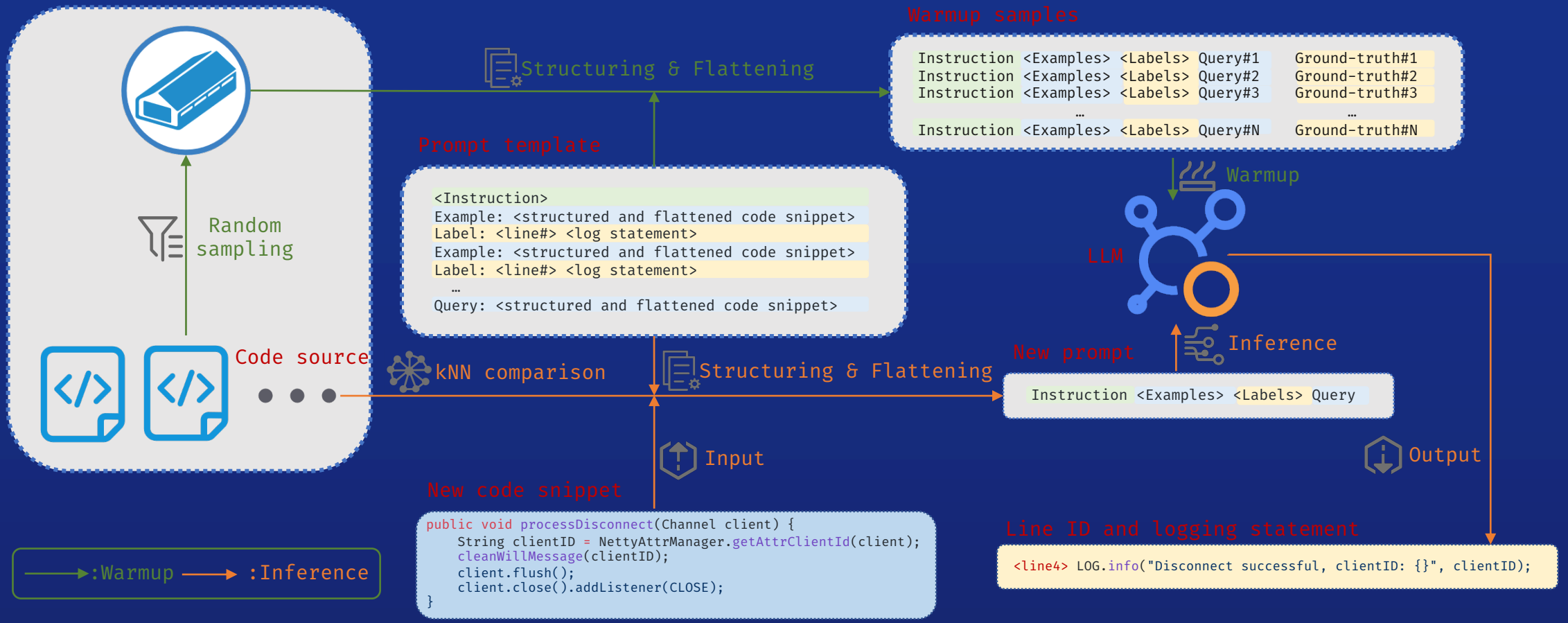
```
<line3> LOG.info(\"Disconnect successful, clientID: {}\", clientID);
```

UniLog：

UniLog是第一个基于上下文学习 (in-context learning, ICL) 的日志记录框架，提供了新的日志记录**范式**：将所有日志记录的细分任务统一构建成一个**文本生成任务**，端到端地实现日志记录。

ICSE'24 UniLog: Automatic Logging via LLM and In-Context Learning

方法实现

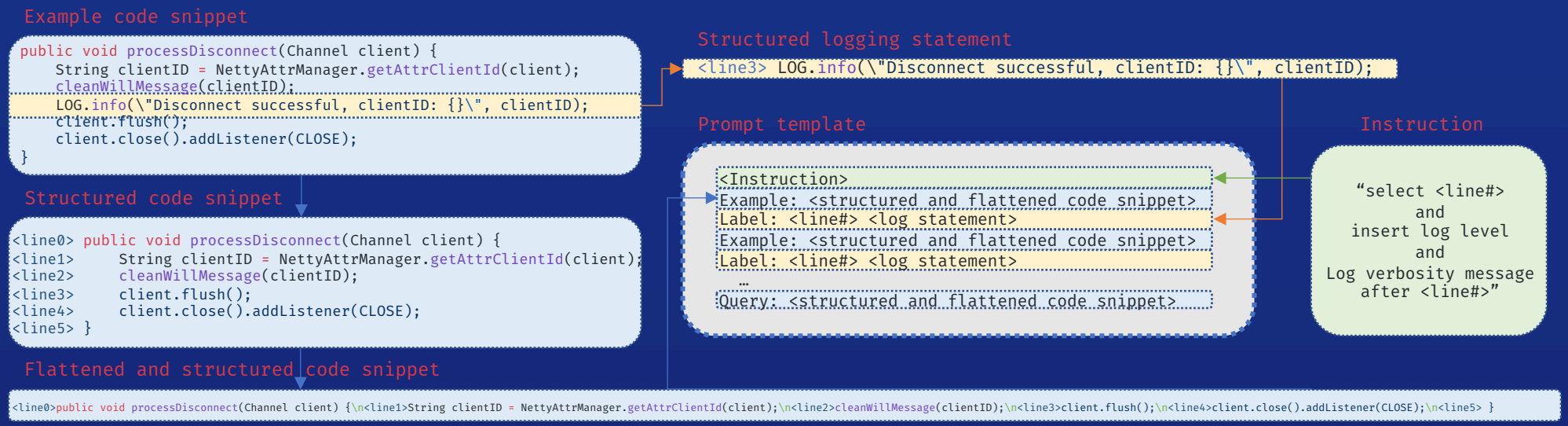


1. UniLog使用行号机制，将日志打印位置的预测任务和日志语句的生成任务统合为文本生成任务

2. UniLog针对每个Query的特征，自动化构建最优Prompt输入模型

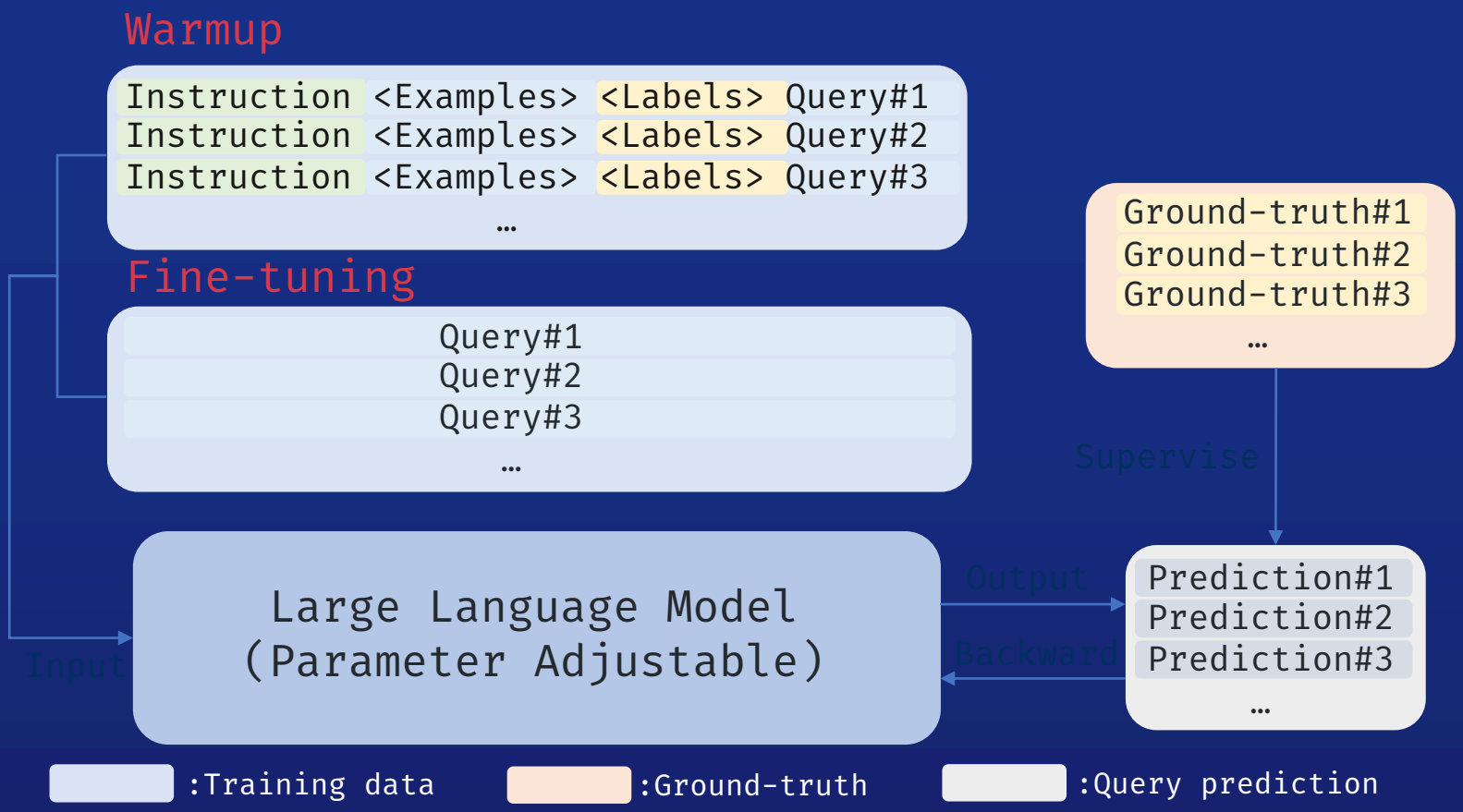
3. UniLog通过随机抽样的ICL样本对模型进行预热，可以进一步提升模型的日志记录精度

方法实现



我们设计了一套自动化流水线来为每个目标代码构建提示输入例和参考输出结果。

方法实现



采用极少量的随机样本对模型进行预热可以进一步提升日志记录性能

实际实验证明预热样本仅需微调完整模型的4%的数据，即可做到比微调效果更好的日志记录表现。

第三章节

评估结果

评估结果

AIOps Challenge

Method	PA	LA	MA	CLA	CMA	BLEU
LANCE-Full	60.2	60.4	13.7	73.2	21.4	N/A
LANCE-Best	65.4	66.2	16.9	76.8	24.3	N/A
UniLog-w/o warmup	66.5	66.8	20.2	75.7	28.7	24.5
UniLog-w/ warmup	76.9	72.3	22.4	77.9	28.1	27.1

UniLog在日志位置、日志等级、日志消息的精度上都远超当前**基于机器翻译范式**的Baselines。

评估结果

AIOps Challenge

```java
public class A {
  protected List<Blob> loadBlobs(List<Map<String, String>> blobInfos) {
    log.debug("Loading blobs from the file system: " + blobInfos);
    List<Blob> blobs = new ArrayList<>();
    for (Map<String, String> info : blobInfos) {
      File blobFile = new File(cacheDir, info.get("file"));
      Blob blob = new FileBlob(blobFile);
      blob.setEncoding(info.get("encoding"));
      blob.setMimeType(info.get("mimetype"));
      blob.setFilename(info.get("filename"));
      blob.setDigest(info.get("digest"));
      blobs.add(blob);
    }
    log.debug("Loaded blobs: " + blobs);
    return blobs;
  }
}
```

: Prediction　　:Ground-truth

有时UniLog会在不同的位置生成不同的日志语句。这些日志语句完全不一样，导致在自动化评估的时候被判定为错例。然而，我们发现部分错例在实际工程中仍然是有意义的。

2023 CCF国际AIOps挑战赛决赛暨“大模型时代的AIOps”研讨会

THANKS