参数的文本分析及其应用

刘 辉

2017/8/6

北京理工大學

报告大纲

- 研究背景
  - 标识符文本的重要性
  - 文本分析与程序语义
  - 实参与形参
  - 关键问题
- 数据获取与数据分析
  - 文本相似性
  - 文本相似性的分布特性
  - 文本相似性与语义关联
- 基于文本相似性的异常参数检测
- 基于文本相似性的参数推荐

} 应用

研究背景

- 静态代码分析
  - 代码优化、软件重构、测试用例生成、缺陷分析、缺陷修复……
- 标识符名称
  - 构成代码的主要成分之一
    - 约三分一
  - 具有丰富的意义信息

```
int height =6；
int width=7；
int Area=height *width;
```

```
int x=6；
int y=7；
int z=x*y;
```

研究背景

- 诺干尝试
  - API Specification（钟浩）
  - Method Specification （Tao Xie）
  - 函数名字与实现的不一致
  - 代码补全
  - 预测程序语法语义特性
  - 检测错误参数
- 标识符文本特性有待发掘
  - 是不是所有标识符名字都有语义信息？
  - 语义相关的标识符，其名字文本是否相似？

参数文本分析

- 实参与形参 （argument & parameter）
  - 实参与形参指向相同的对象，语义紧密相关

①实参与形参具有较高的文本相似性。

②参数的文本特性对有助于改善现有的代码分析技术。

参数文本分析

- RQ1：实参与形参在文本上是否相似？有多相似？
- RQ2：实参与形参的本文相似性是否与参数的长度有关？
- RQ3： 为什么有些实参与形参文本上并不相似？
- RQ4：如果某个形参在项目A上和实参不相似，那么在其他项目上同名的形参也和实参不相似么？
- RQ5：与其他候选参数相比，正确的参数是否与形参更为相似？
- RQ6：文本相似度的计算方法对以上结果有多大的影响？
- RQ7：不同的编程语言，其参数文本特性是否近似？
- RQ8：就参数文本相似性而言， primitive 和 non-primitive的parameter是否有本质区别？
- RQ9：就参数文本相似性而言， API 和 non-API 的parameter是否有本质区别？

报告大纲

- 研究背景
  - 标识符文本的重要性
  - 文本分析与程序语义
  - 实参与形参
  - 关键问题
- 数据获取与数据分析
  - 文本相似性
  - 文本相似性的分布特性
  - 文本相似性与语义关联
- 基于文本相似性的异常参数检测
- 基于文本相似性的参数推荐

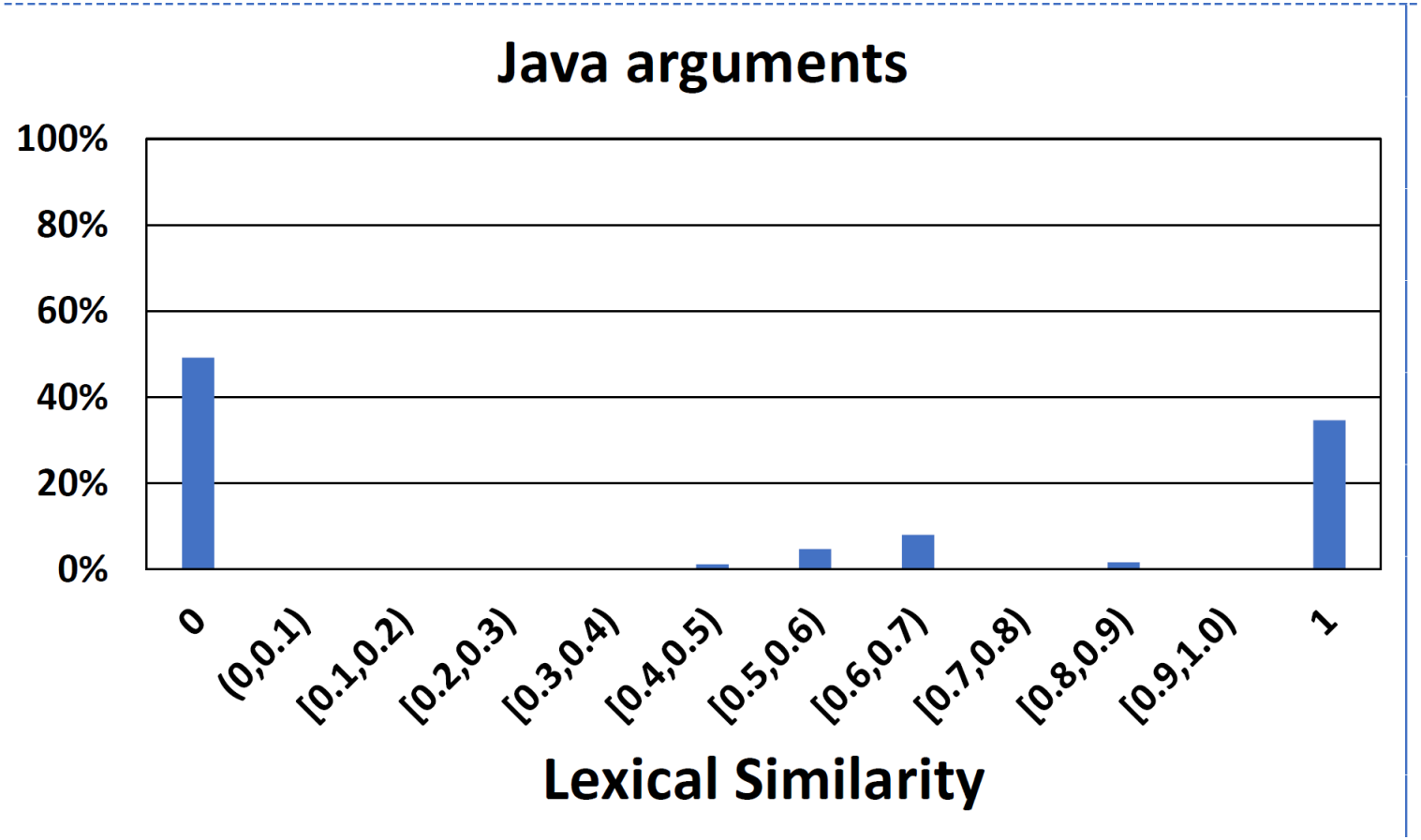
数据获取

- 来源
  - Source Forge、Github
- 项目数
  - 120个Java项目、25个C语言项目
- 规模
  - Java：一千八百万行
  - C：五百万行
- 关键数据
  - 有名字的实参 & 形参
  - 实参数：90万+25万

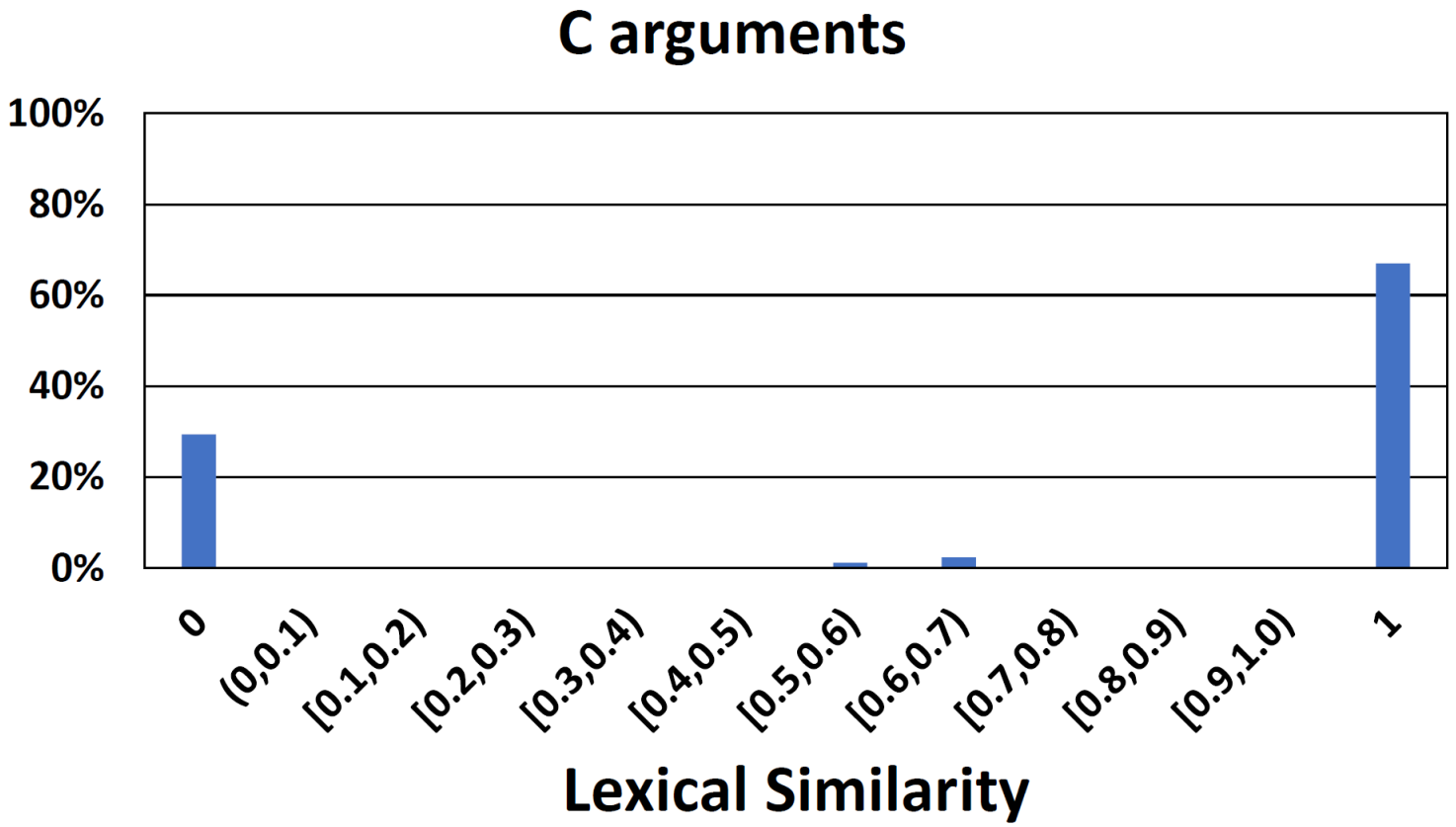
相似度计算

- 基于相同的单词数
  - Sim（abac，ac）=(3+2)/(4+2)
- 基于编辑距离
  - Sim（abc，aec）=1-1/3
- 基于相同字母及其顺序
  - JaroWinkler-based metrics

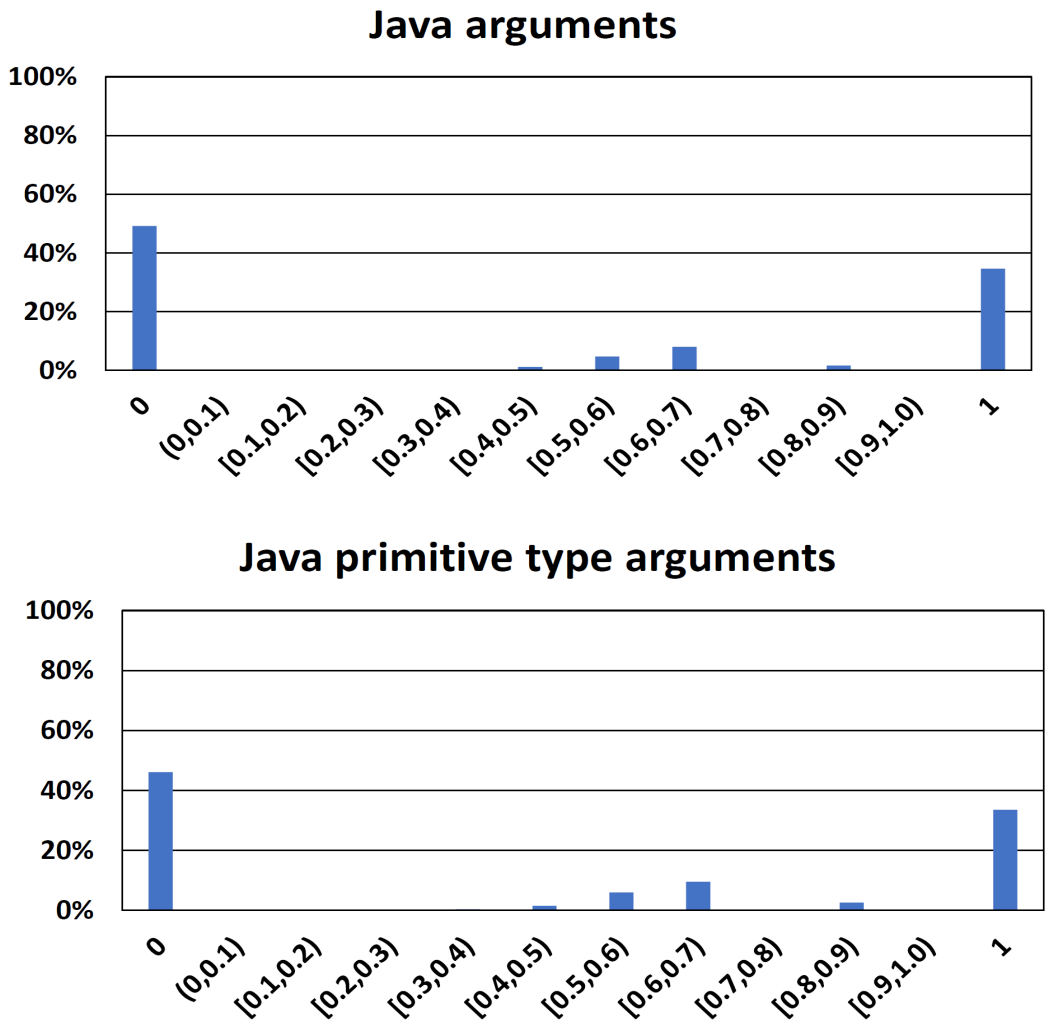
RQ1：实参与形参的文本相似性分布特征



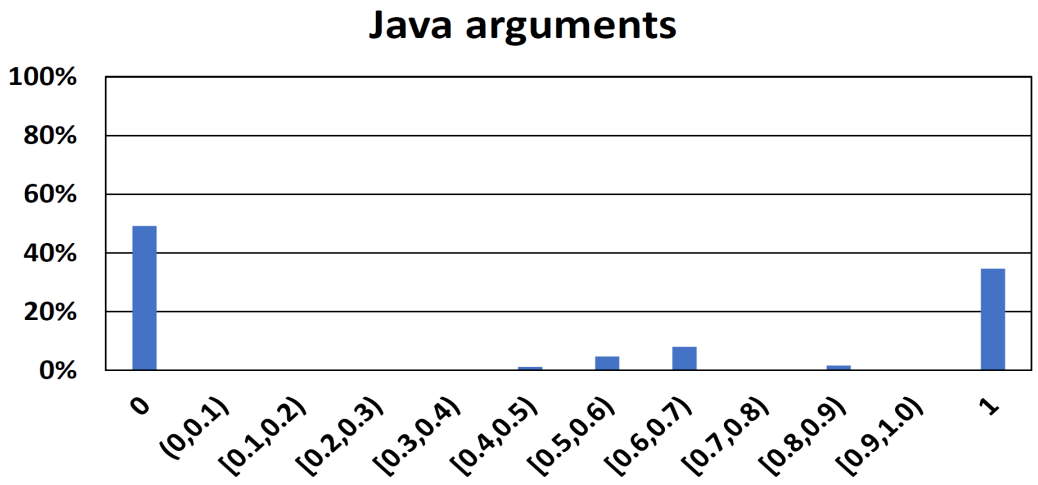
RQ1：实参与形参的文本相似性分布特征

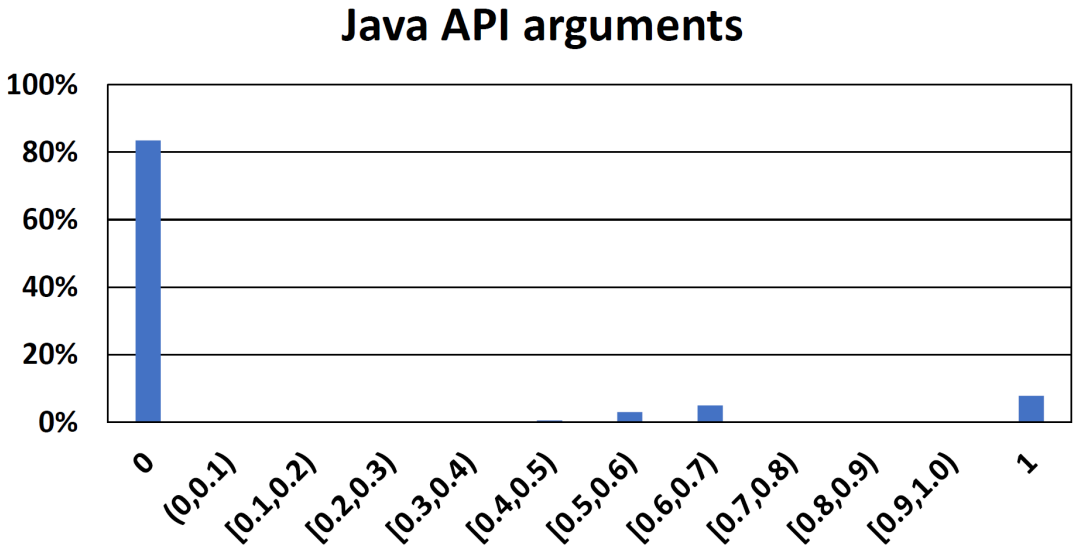


RQ1：实参与形参的文本相似性分布特征

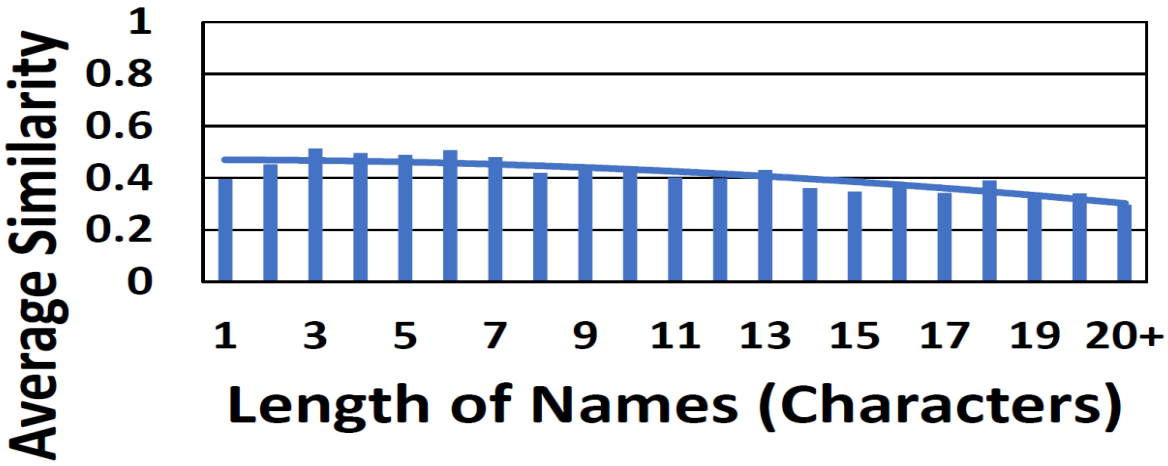


RQ1：实参与形参的文本相似性分布特征

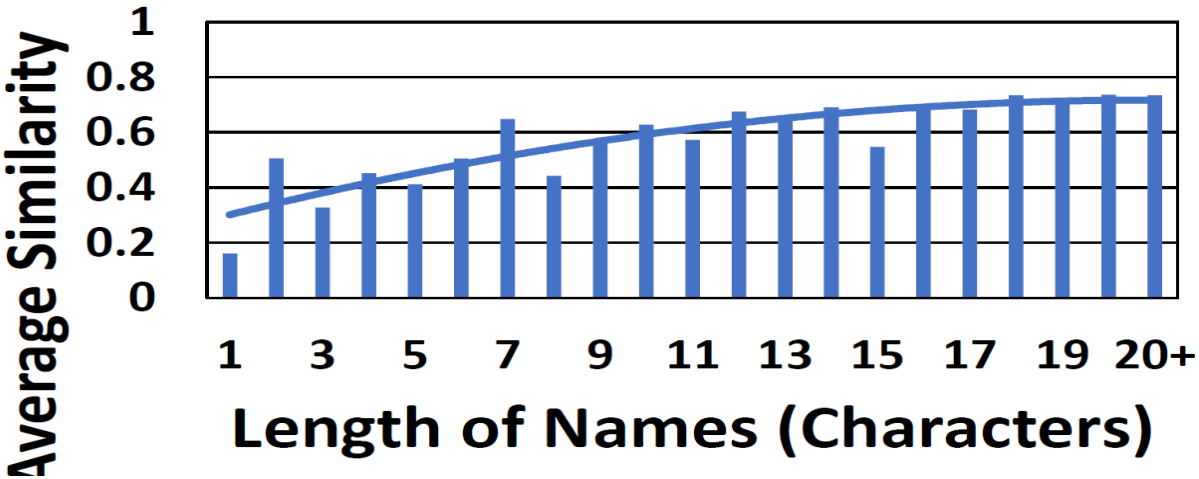




RQ2：Length vs Similarity

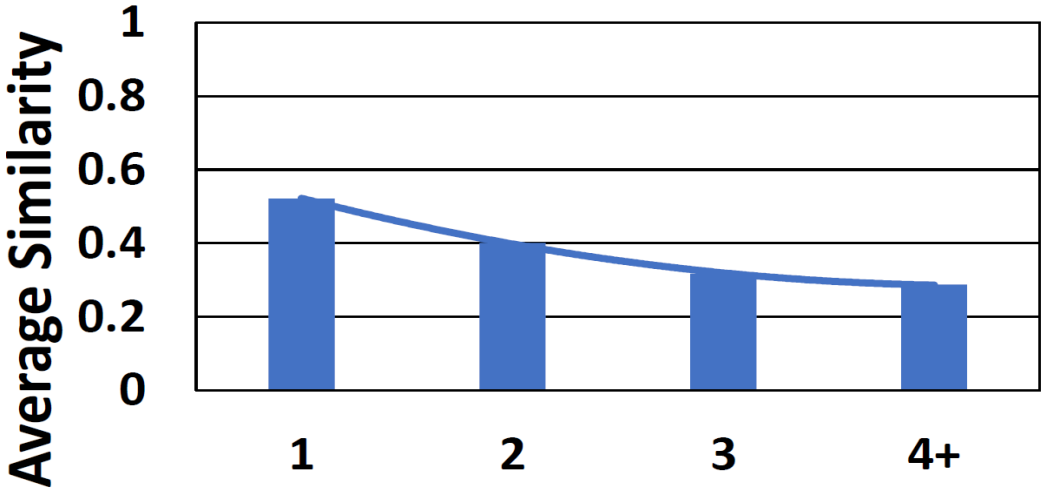


实参

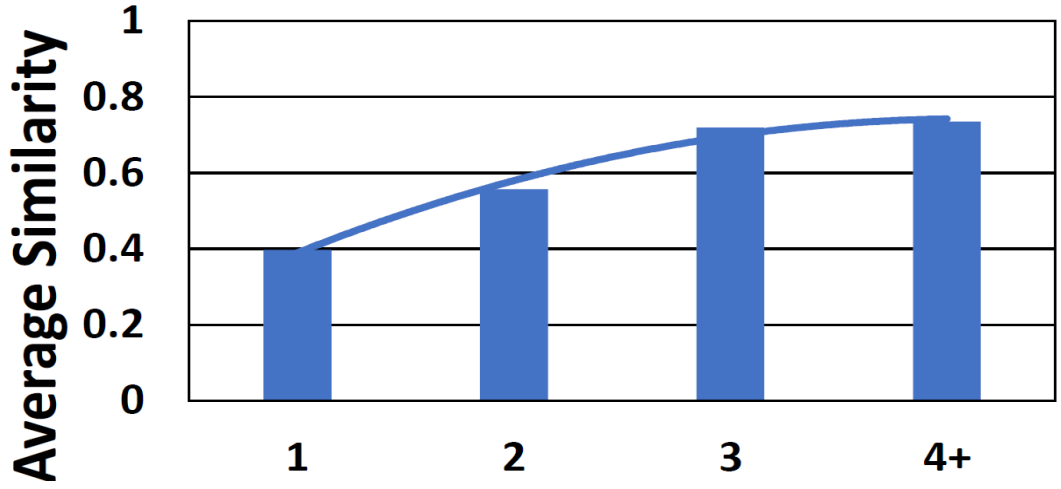


形参

RQ2：Length vs Similarity

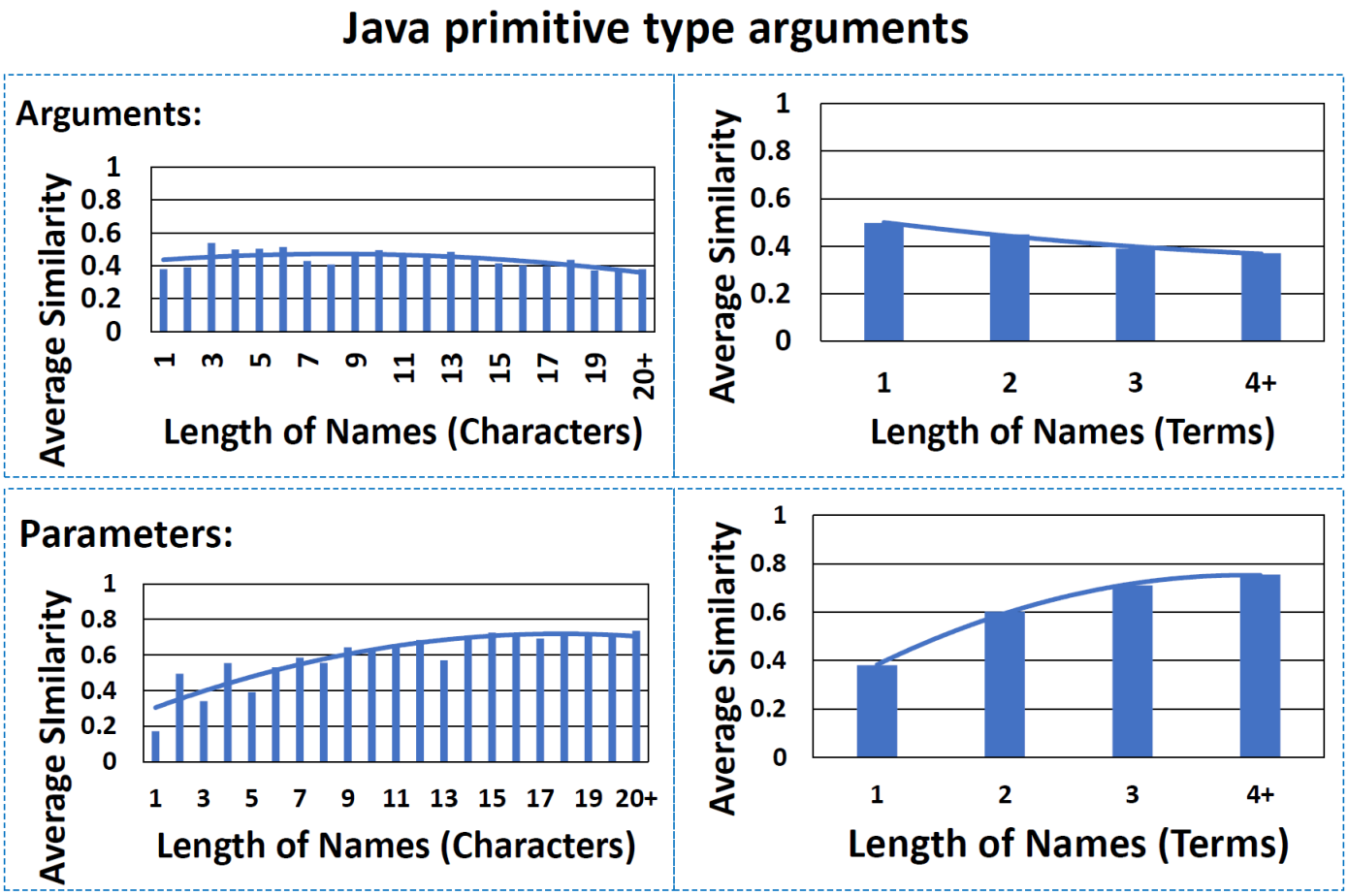


实参

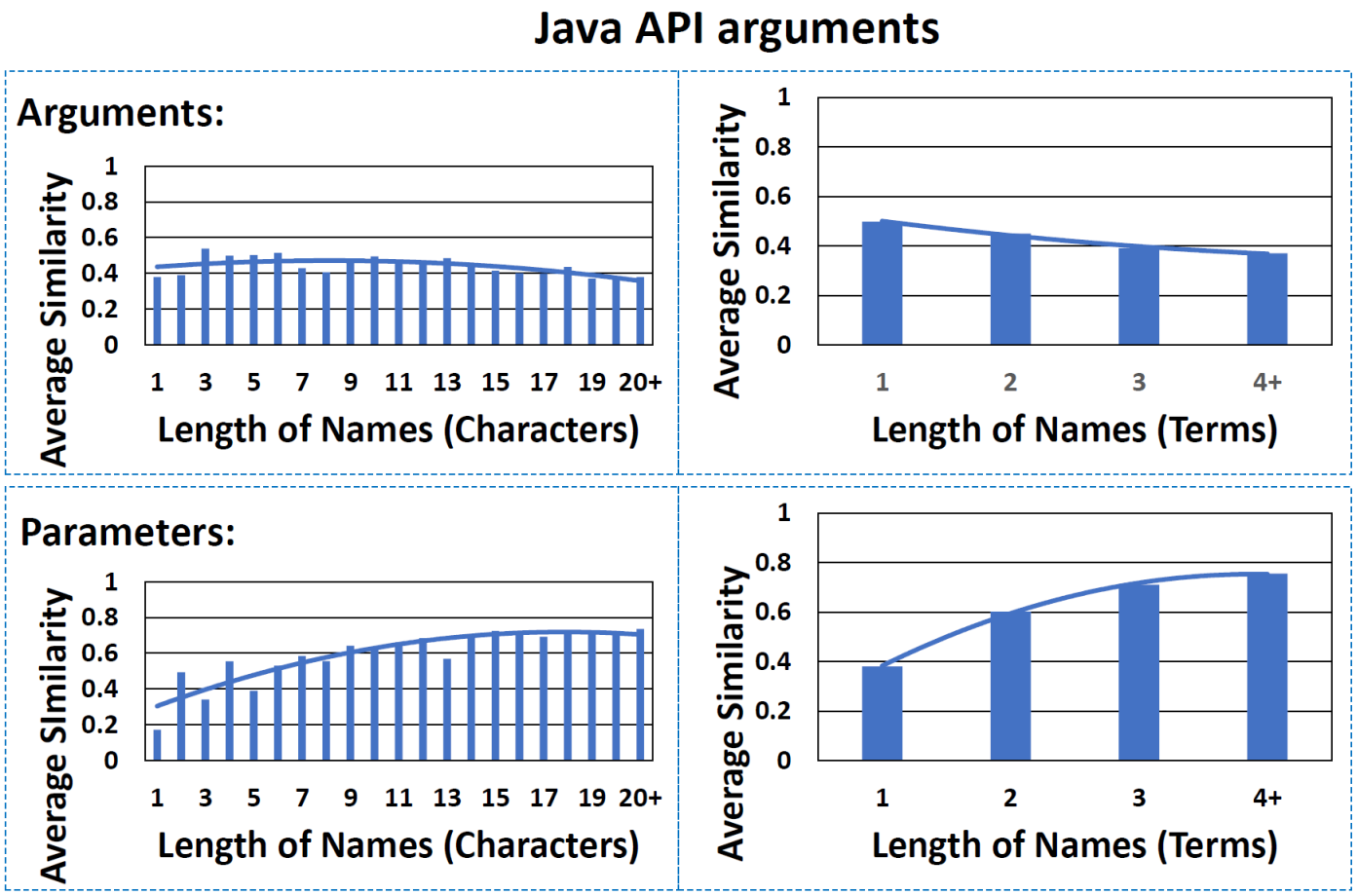


形参

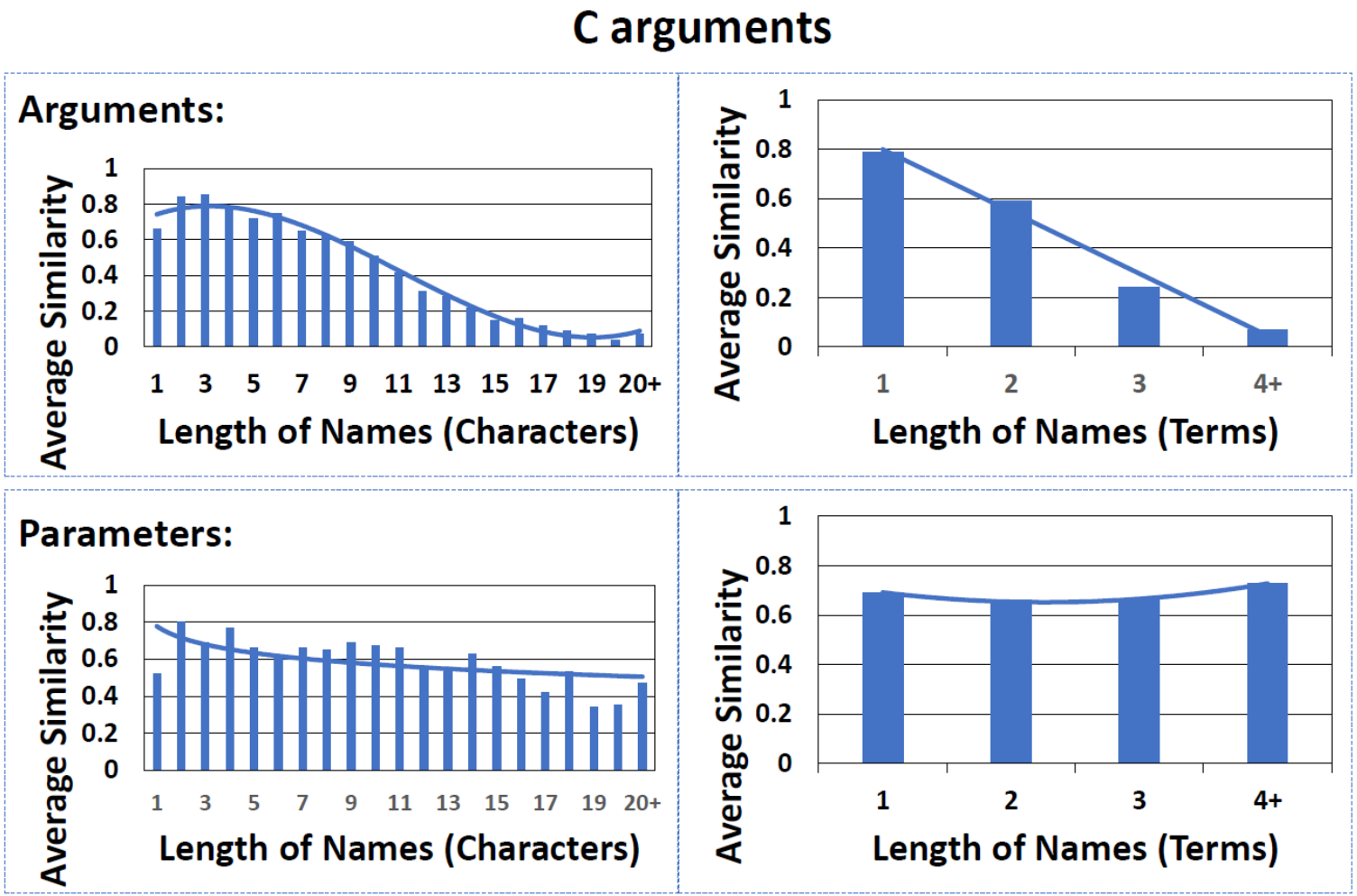
RQ2 : Length vs Similarity



RQ2 : Length vs Similarity



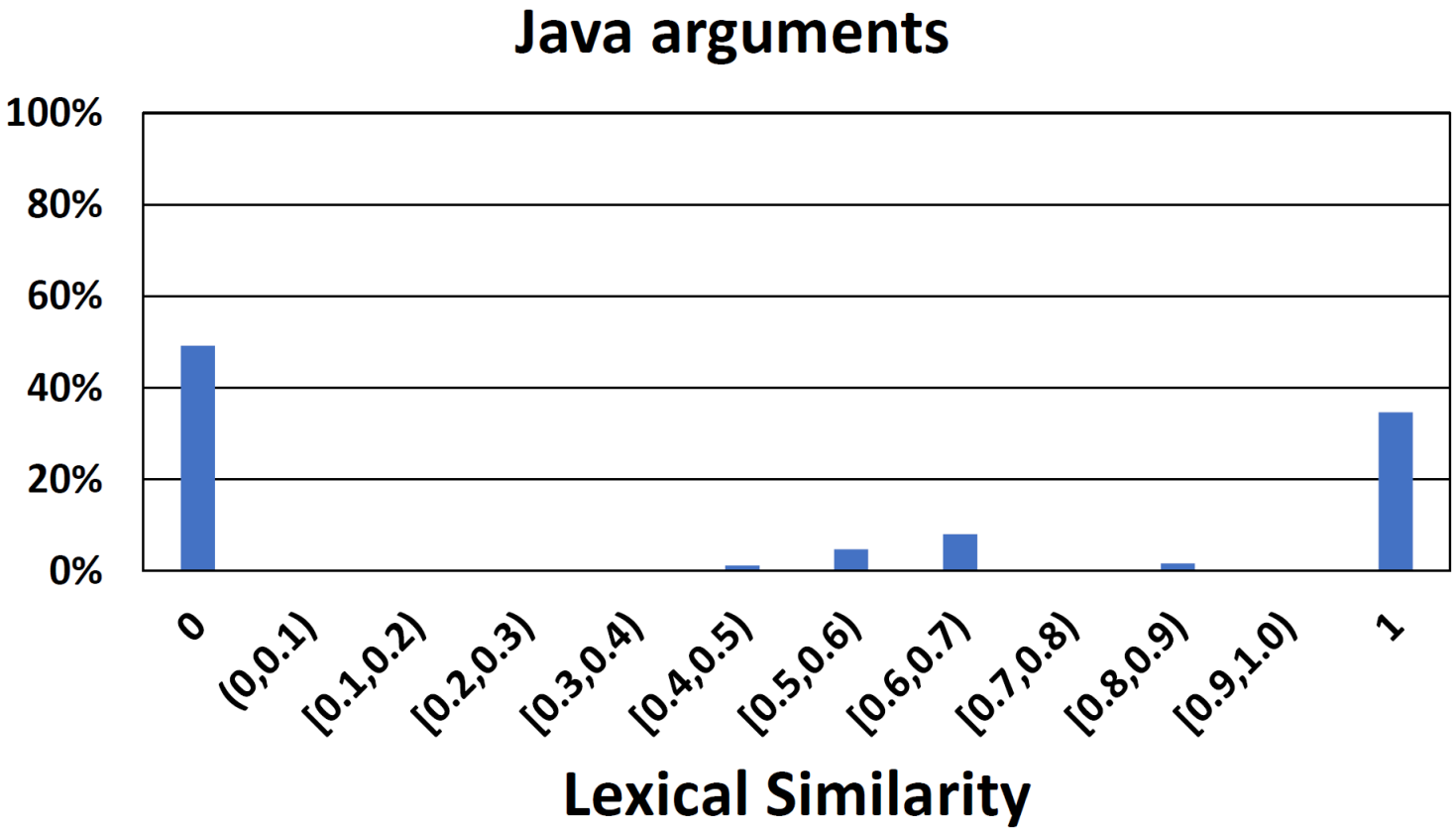
RQ2 : Length vs Similarity



RQ2：Length vs Similarity

- 相关系数偏低
  - 0.27 (parameters) and -0.11 (arguments) for Java arguments.
  - -0.06 (parameters) and -0.04 (arguments) for C arguments
- 原因
  - 相同长度的参数，其相似性差异巨大

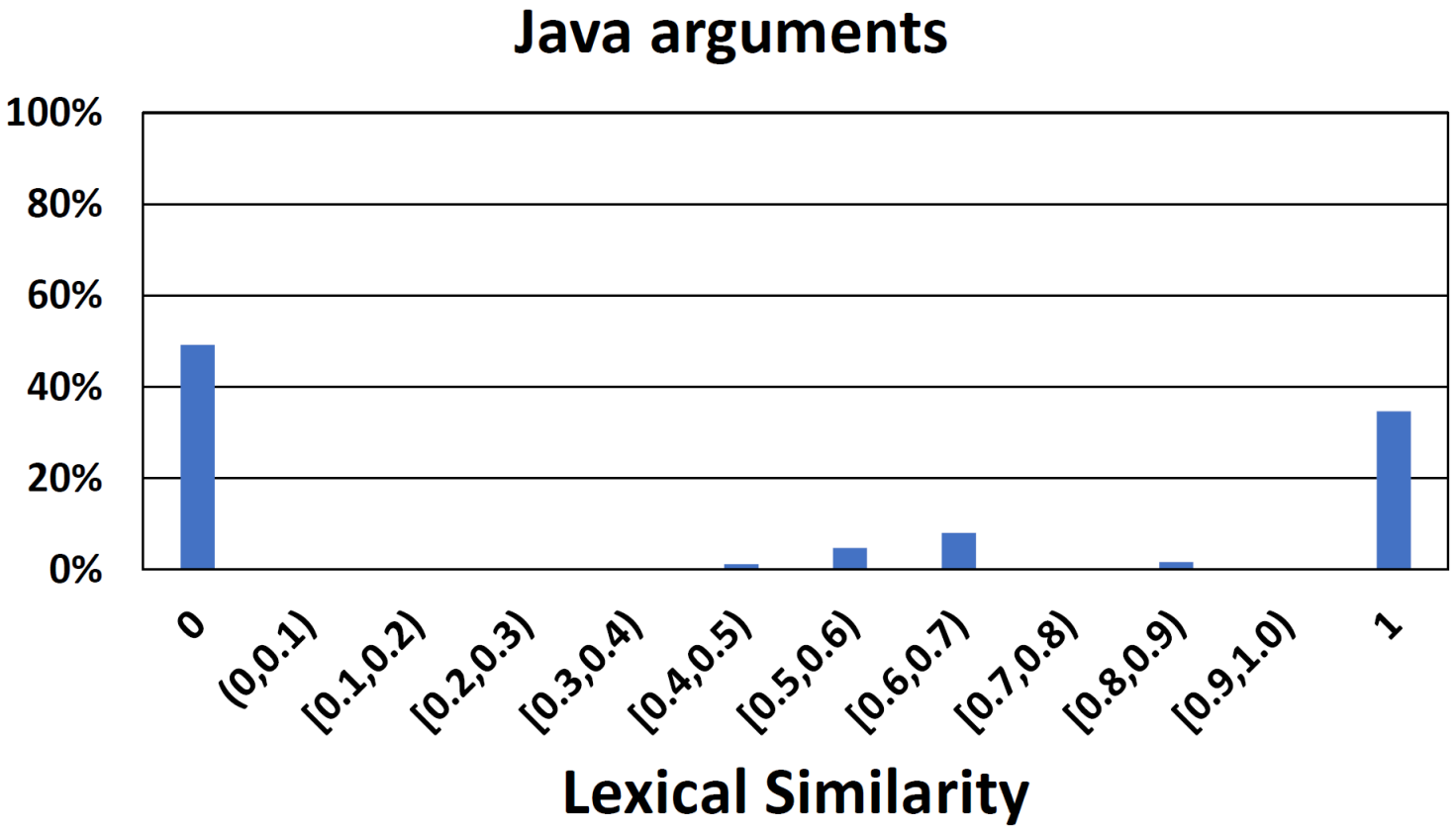
RQ3：为何不相似？


Java arguments
100%
80%
60%
40%
20%
0%
0
(0,0.1)
[0.1,0.2)
[0.2,0.3)
[0.3,0.4)
[0.4,0.5)
[0.5,0.6)
[0.6,0.7)
[0.7,0.8)
[0.8,0.9)
[0.9,1.0)
1
Lexical Similarity

RQ3：为何不相似？

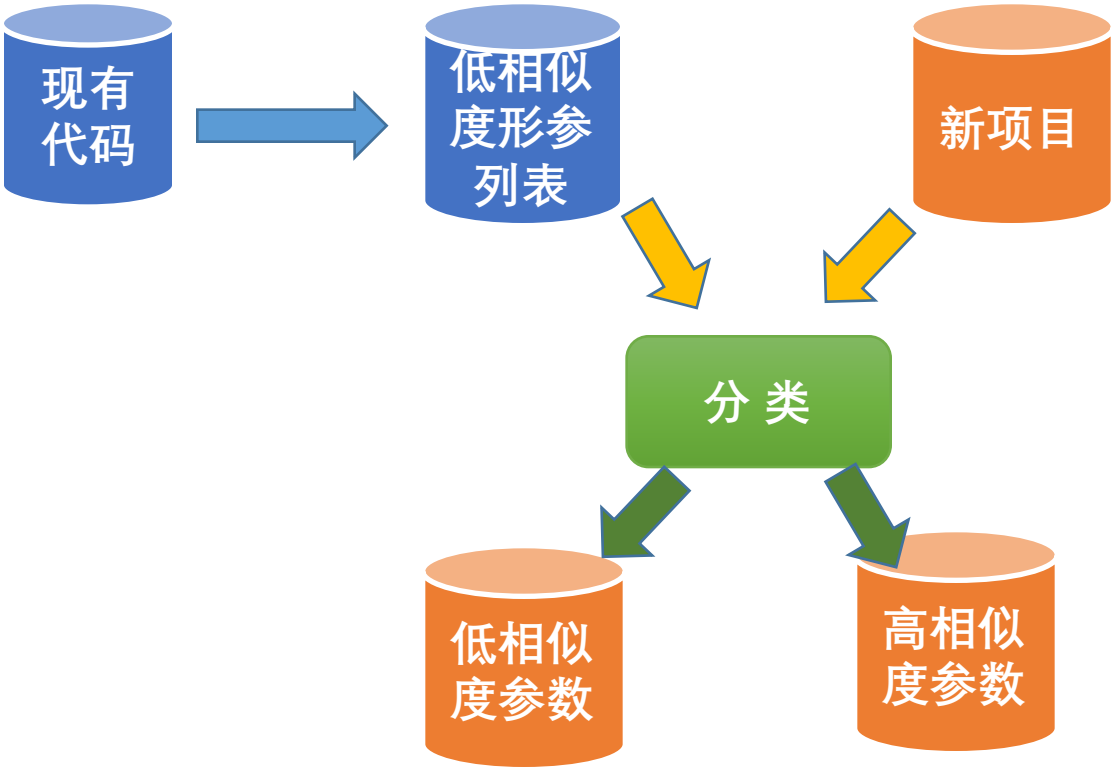
- 抽样200个实参，手工分析
- 主要原因
  - 超短名称
    - 30%: 形参名只包含1个字母
    - 53%：形参名不多于3个字母.
  - 集合操作
    - 8.5%：index, item, key, value
- 整个数据集上进行验证
  - SQL查询
  - 结论基本一致

RQ4：过滤低相似度参数



RQ4：过滤低相似度参数

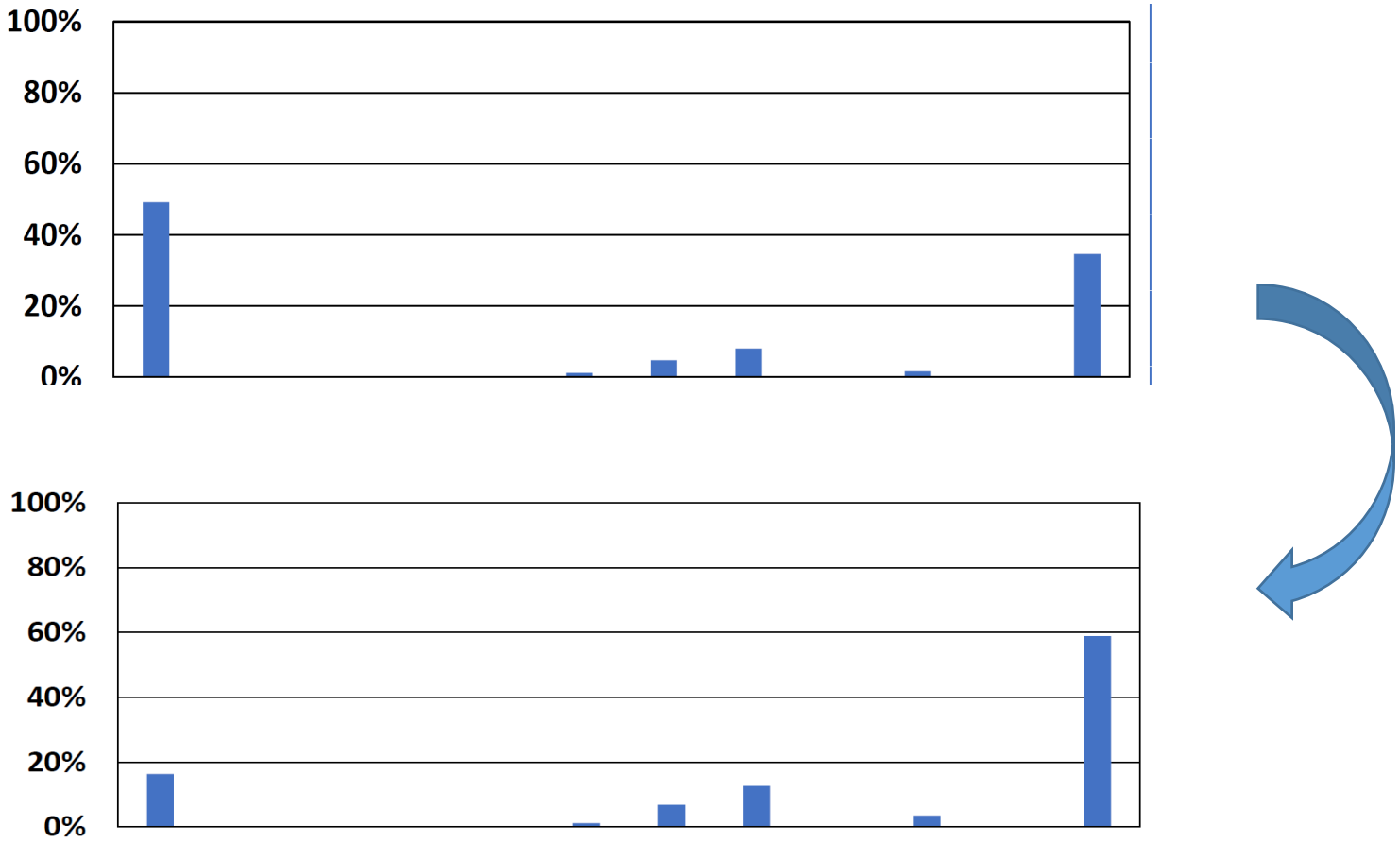
- 低相似度参数（形参）
  - 在给定数据集（如某个项目）上，该形参与实参的平均相似度小于0.5



RQ4：过滤低相似度参数

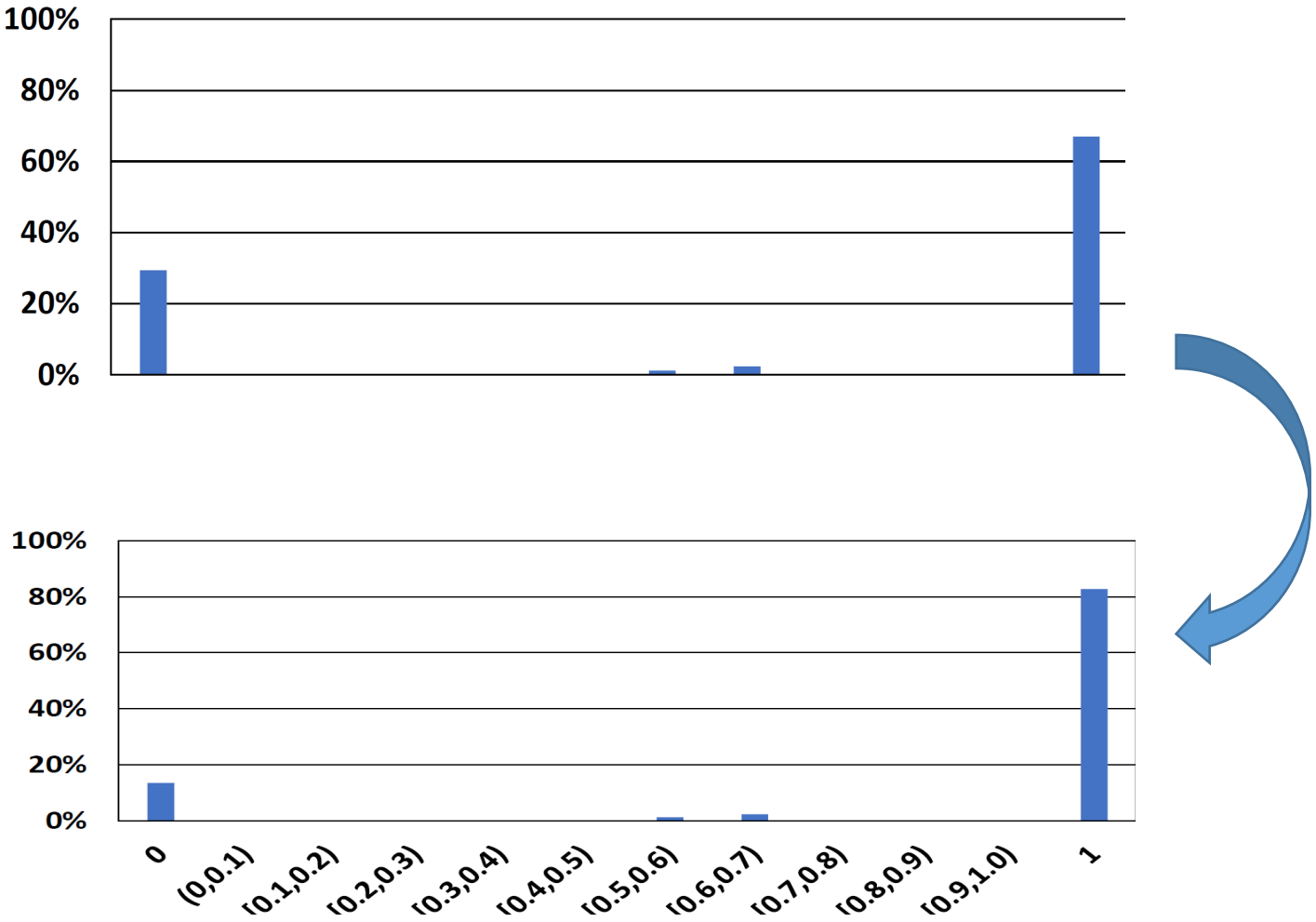
Similarity	Arguments (n_1)	Filtered out arguments (n_2)	n_2/n_1
[0.0, 0.1)	426,445	362577	85%
[0.1, 0.2)	16	4	25%
[0.2, 0.3)	1,077	785	73%
[0.3, 0.4)	2,845	1,701	60%
[0.4, 0.5)	10,279	5,483	53%
[0.5, 0.6)	40,721	13,701	34%
[0.6, 0.7)	69,422	19,980	29%
[0.7, 0.8)	398	13	3%
[0.8, 0.9)	14,779	1,136	8%
[0.9, 1.0)	66	3	5%
1	30,0189	69,944	23%
Total	866,237	475,327	55%

RQ4：过滤低相似度参数



RQ4：过滤低相似度参数

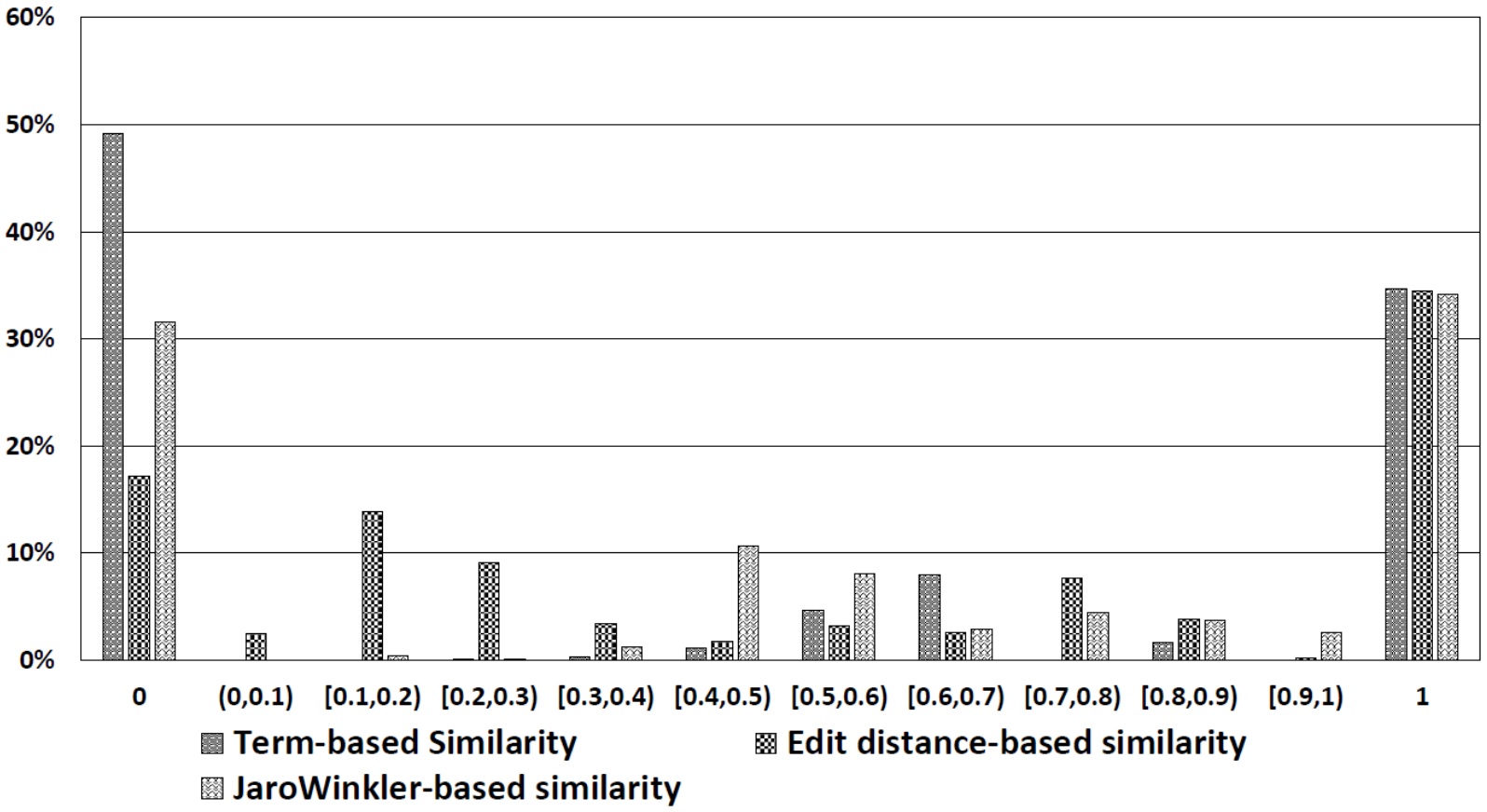
C



RQ5：正确参数VS候选参数

- 候选参数：语法正确的错误参数
- 40% Java 参数有其他候选参数可用
- 15% ：候选参数比正确参数具有更高的文本相似性
- 94%：MinSim=2/3

RQ6：相似度计算方法的影响



报告大纲

- 研究背景
  - 标识符文本的重要性
  - 文本分析与程序语义
  - 实参与形参
  - 关键问题
- 数据获取与数据分析
  - 文本相似性
  - 文本相似性的分布特性
  - 文本相似性与语义关联
- 基于文本相似性的异常参数检测
- 基于文本相似性的参数推荐

基于文本相似性的异常参数检测

$$lexSim(m\_alt, par) - lexSim(curArg, par) \geq \beta$$

- 基于ChangeDistiller 获取所有只影响一个实参的change
- 手动检查确认错误参数
  - 14个错误参数
  - 60个项目
- 169个警报，包括9个错误参数、127个rename opportunities、33个误报
  - R=64%
  - P=80%

报告大纲

- 研究背景
  - 标识符文本的重要性
  - 文本分析与程序语义
  - 实参与形参
  - 关键问题
- 数据获取与数据分析
  - 文本相似性
  - 文本相似性的分布特性
  - 文本相似性与语义关联
- 基于文本相似性的异常参数检测
- 基于文本相似性的参数推荐

基于文本相似性的参数推荐

- 输入：函数f、形参p
- 输出：实参a

1. If p∈LowSimPar，拒绝推荐
2. 找出所有候选实参（语法正确）
3. 计算候选实参与p的文本相似性
4. 找出具有最好相似度的候选实参a
5. If sim（a，p）< minSim，拒绝推荐
6. 推荐a为实参

基于文本相似性的参数推荐

Application	Size (LOC)	Recommended Arguments	Precision
Neuroph	11,377	326	80%
WURFL	10,252	343	87%
Json-lib	8,055	122	92%
Joda-Time	27,779	797	81%
Total	57,463	1,588	83%

谢谢！