面向任务的代码搜索

江贺 (jianghe@dlut.edu.cn)

聂黎明 (limingnie@mail.dlut.edu.cn)

大连理工大学

合作者: 孙泽义 任志磊 李晓晨 孔维强 张涛 罗夏朴

代码搜索

在软件开发过程中，代码搜索能够为开发者提供参考的代码段去辅助完成特定的编程任务。

Section One

基础数据和影响力分析

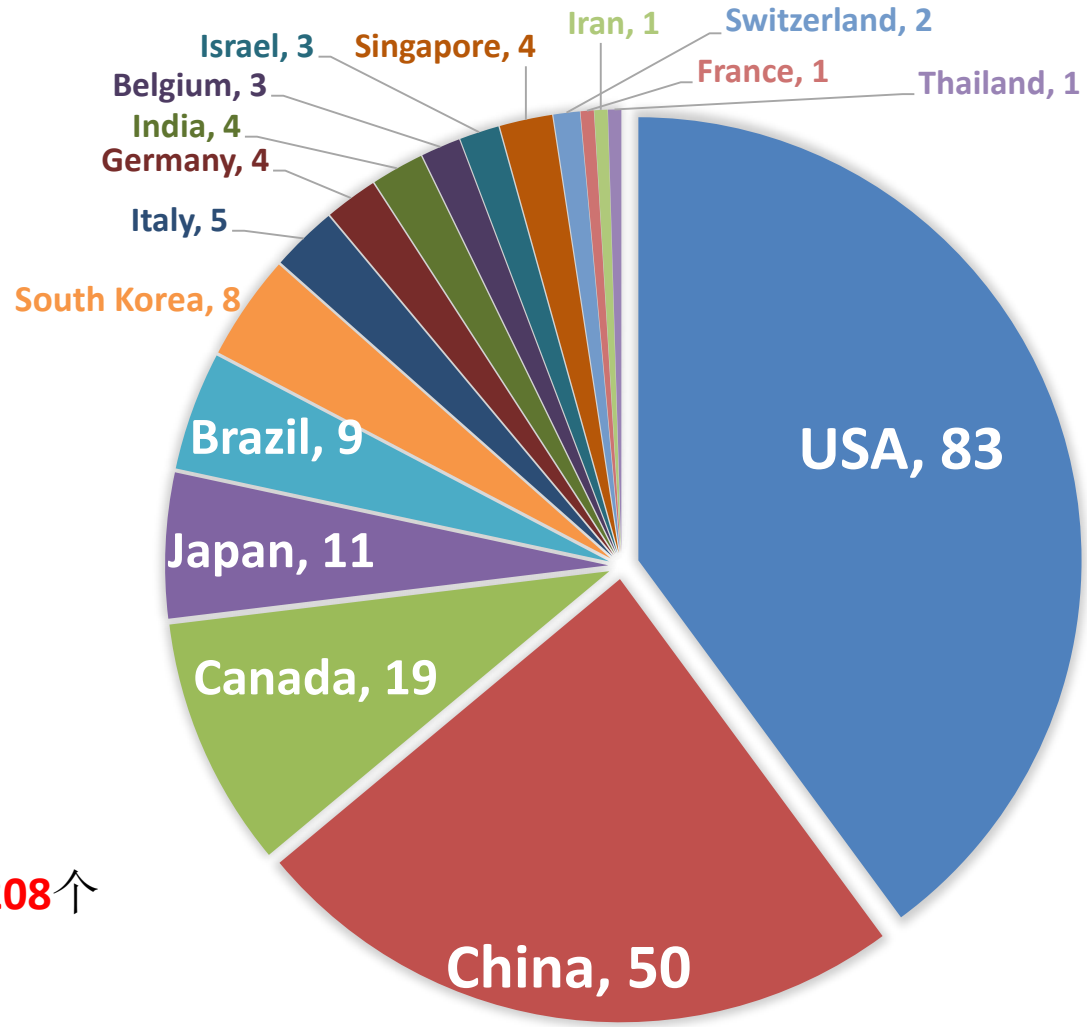
By 江贺

基础数据和影响力分析

- 收集了**93篇**与代码/API 推荐相关的文献
- 作者总计**208**个，国家**16**个。

- 分析回答以下问题：
- 1. Where? 作者来自哪里？
- 2. Who? 谁是最高产的作者？影响力如何?
- 3. Which? 那篇文章被引次数最多？
- 4. Co-authorship network 合著网络？

作者来自哪里？



作者总计**208**个
国家**16**个

谁是最高产的作者？

序号	作者	相关论文篇数	H-index	国家
1	David Lo	**10**	33	Singapore
2	Collin McMillan	**8**	11	USA
3	Denys Poshyvanyk	**8**	39	USA
4	Shaowei Wang	**8**	11	Singapore
5	Mark Grechanik	**7**	20	USA
6	Emily Hill	**6**	18	USA
7	Tao Xie	**4**	49	USA
8	Reid Holmes	**4**	20	Canada
9	Sushil Bajracharya	**4**	20	USA
10	Cristina Lopes	**4**	40	USA

谁是最高产的作者？

序号	作者	相关论文篇数	H-index	国家
1	David Lo	**10**	33	Singapore
2	Collin McMillan	**8**	11	USA
3	Denys Poshyvanyk	**8**	39	USA
4	Shaowei Wang	**8**	11	Singapore
5	Mark Grechanik	**7**	20	USA
6	Emily Hill		18	USA
7	Tao Xie		49	USA
8	Reid Holmes		20	Canada
9	harya		20	USA
10	pes		40	A

作者影响力？

序号	作者	相关论文篇数	ACS指数	总被引次数
1	Tao Xie	4	**245.3**	723
2	Reid Holmes	4	**184.8**	411
3	Gail C. Murphy	2	**165.7**	350
4	Steven P. Reiss	1	**165.0**	165
5	Suresh Thummalapenta	2	**156.0**	309
6	K. Vijay-Shanker	3	**121.6**	483
7	Naiyana Sahavechaphan	1	**100.5**	201
8	Kajal Claypool	1	**100.5**	201
9	David Mandelin	1	**87.5**	350
10	Lin Xu	1	**87.5**	350

注: ACS不仅衡量了作者对文章的贡献度，还衡量了对该领域的贡献度

作者影响力？

序号	作者	相关论文篇数	ACS指数	总被引次数
1	Tao Xie	4	**245.3**	723
2	Reid Holmes	4	**184.8**	411
3	Gail C. Murphy	2	**165.7**	350
4	Steven P. Reiss	1	**165.0**	165
5	Suresh Thummalapenta	2	**156.0**	309
6	K. Vijay-Shanker		**121.6**	483
7	Naiyana Sahavechaphan		**100.5**	201
8	Kaj[illegible]	1	**100.5**	201
9	Dav[illegible]	1		350
10		1		350

注: ACS不仅衡量[illegible]文章的贡献度，还衡量[illegible]域的贡献度

哪篇文章被引数最多？

序号	题目	被引次数	NCII	期刊
1	Jungloid mining: helping to navigate the API jungle	**350**	31.82	PLDI 2005
2	Parseweb: a programmer assistant for reusing open source code on the web	**306**	34.00	ASE 2007
3	Using structural context to recommend source code examples	**294**	26.73	ICSE 2005
4	XSnippet: mining For sample code	**201**	20.10	OOPSLA 2006
5	Mining API patterns as partial orders from source code: from usage scenarios to specifications	**201**	22.33	FSE 2007
6	Using natural language program analysis to locate and understand action-oriented concerns	**182**	20.22	AOSD 2007
7	MAPO: Mining and Recommending API Usage Patterns	**180**	25.71	ECOOP 2009
8	Example-centric programming: integrating web search into the development environment	**173**	28.83	CHI 2010
9	Semantics-based code search	**165**	23.57	ICSE 2009
10	Learning from examples to improve code completion systems	**155**	22.14	FSE2009

哪篇文章被引数最多？

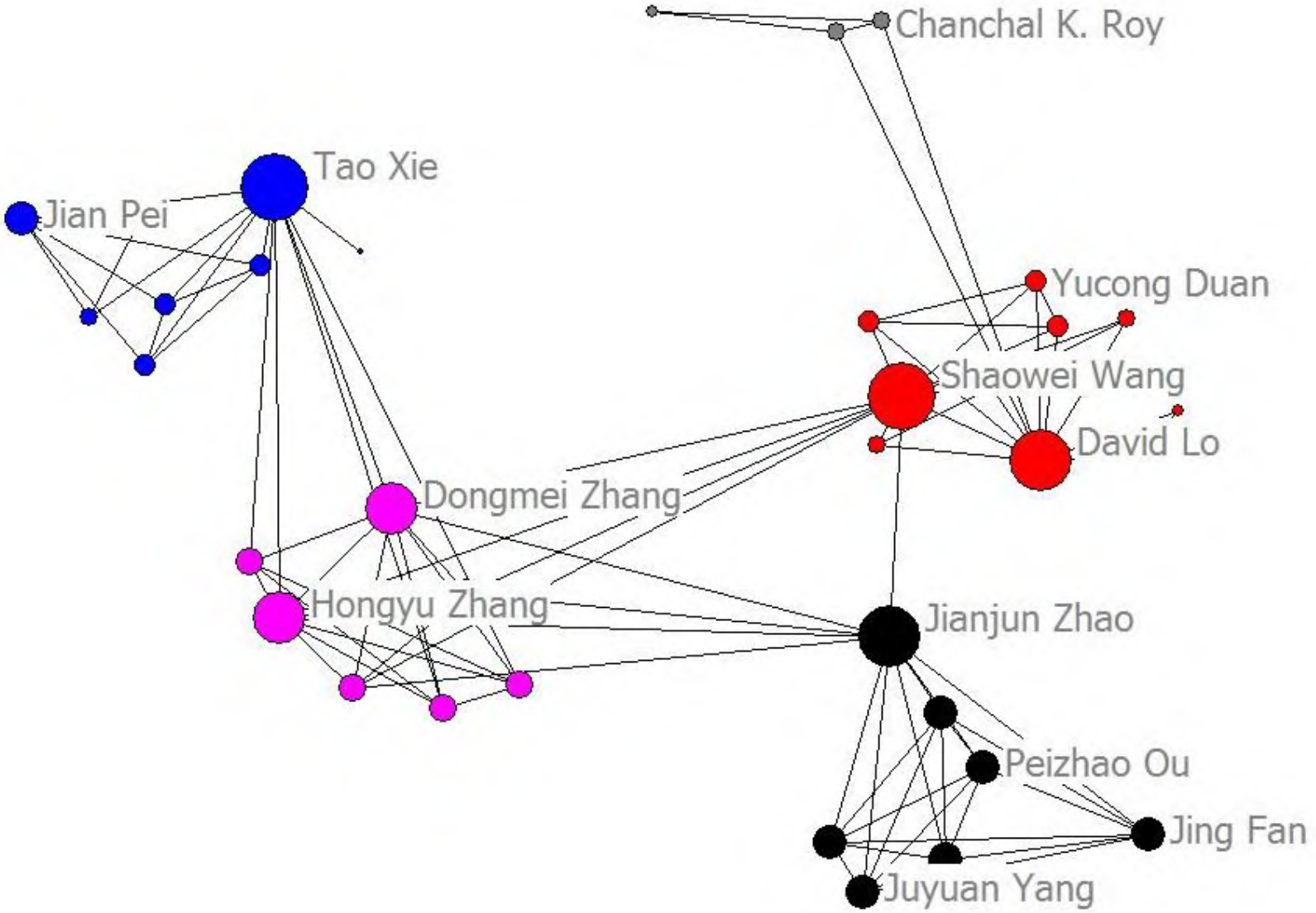
序号	题目	被引次数	NCII	期刊
1	Jungloid mining: helping to navigate the API jungle	350	31.82	PLDI 2005
2	Parseweb: a programmer assistant for reusing open source code on the web	306	34.00	ASE 2007
3	Using structural context to recommend source code examples	294	26.73	ICSE 2005
4	XSnippet: mining For sample code	201	20.10	OOPSLA 2006
5	Mining API patterns as partial orders from source code: from usage scenarios to specifications	201	22.33	FSE 2007
6	Using natural language program analysis to locate and understand action-oriented concerns	182	20.22	AOSD 2007
7	Static object ranking for programming by example	180	25.70	ECOOP 2009
8	Example-centric programming: integrating web search into the development environment	173	28.83	CHI 2010
9	Semantics-based code search	165	23.57	IEEE S&P 2009
10	Learning from examples to improve code completion systems	155	22.14	FSE2009

1.Jungloid mining: helping to navigate the API jungle, PLDI 2005.

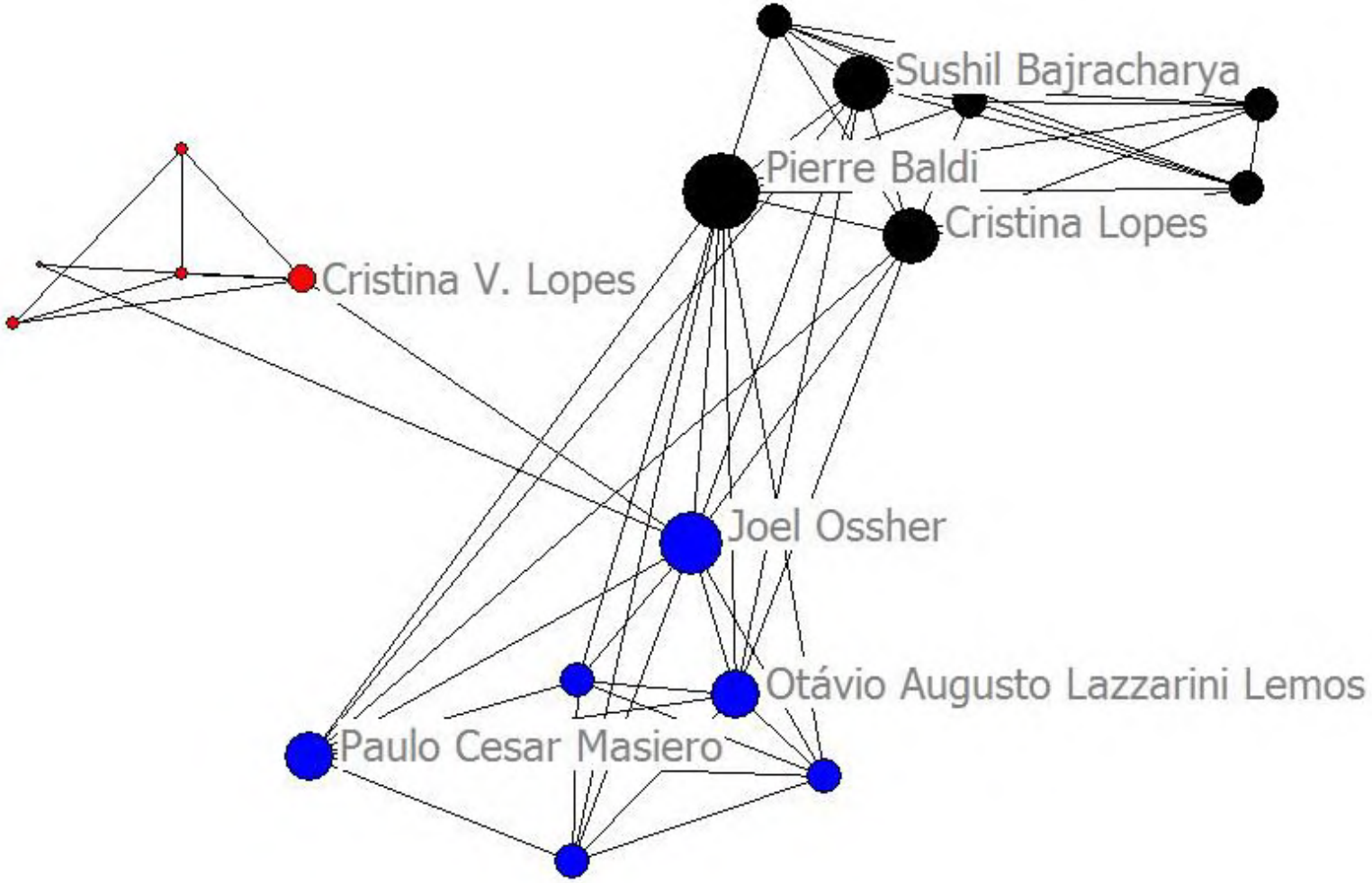
2.Parseweb: a programmer assistant for reusing open source code on the web, ASE 2007.

3.Using structural context to recommend source code examples,ICSE 2005.

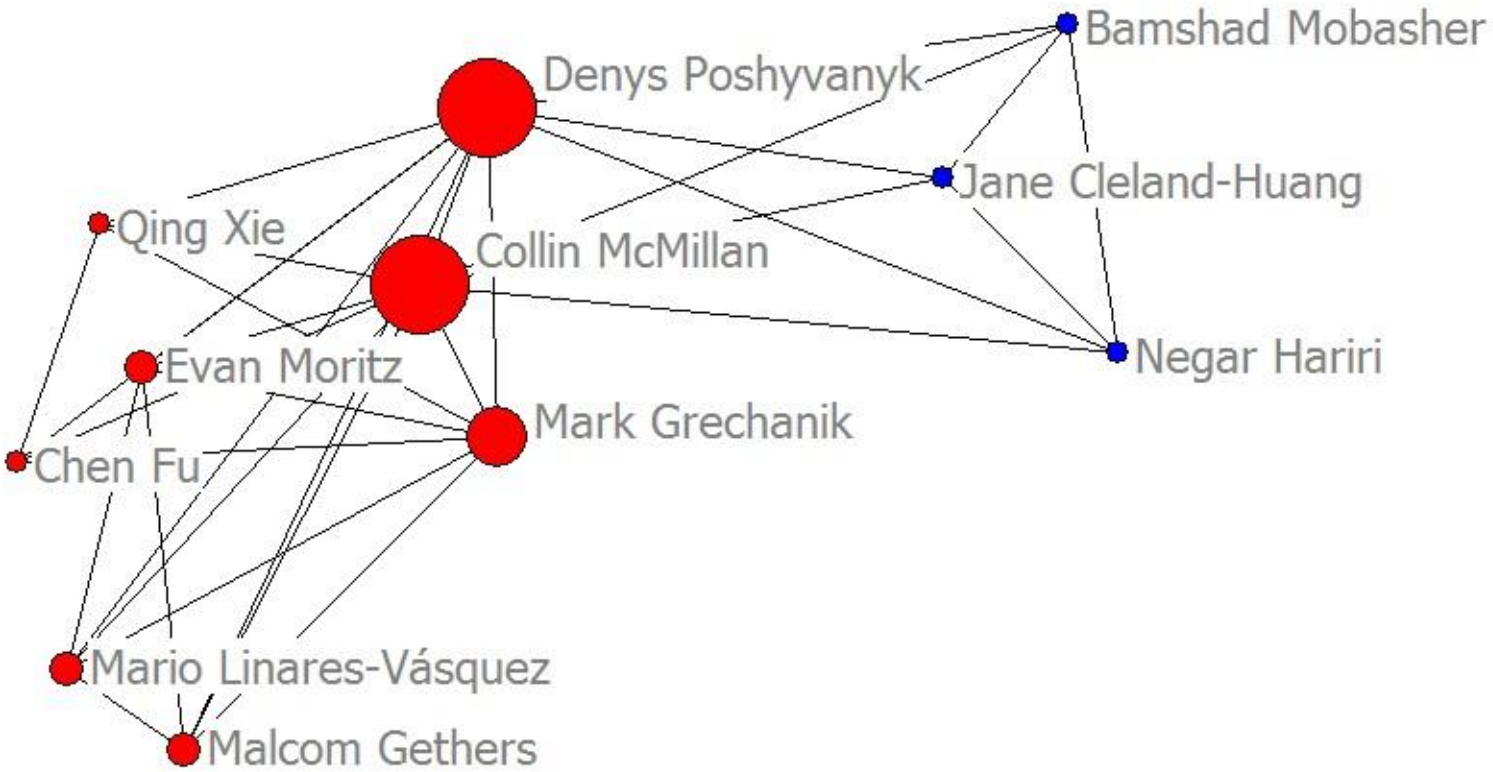
作者合著网络（1）


Chanchal K. Roy
Tao Xie
Jian Pei
Yucong Duan
Shaowei Wang
David Lo
Dongmei Zhang
Hongyu Zhang
Jianjun Zhao
Peizhao Ou
Jing Fan
Juyuan Yang

作者合著网络（2）


Sushil Bajracharya
Pierre Baldi
Cristina Lopes
Cristina V. Lopes
Joel Ossher
Otávio Augusto Lazzarini Lemos
Paulo Cesar Masiero

作者合著网络（3）


Bamshad Mobasher
Denys Poshyvanyk
Jane Cleland-Huang
Qing Xie
Collin McMillan
Negar Hariri
Evan Moritz
Mark Grechanik
Chen Fu
Mario Linares-Vásquez
Malcom Gethers

Section Two

代码搜索的两个尝试

By 聂黎明

代码搜索的分类

按输入类型分：

自由文本作为查询（面向任务的代码搜索）

- **Lv et al. ASE 2015, Keivanloo et al. ICSE 2014, McMillan et al. TOSEM 2013, Bajracharya et al. FSE 2010.**

API作为查询

- **Subramanian et al. ICSE 2014, Ghafari et al. ICPC 2014, Moritz ASE 2013, Wang et al. ASE 2011. Zhong et al. ECOOP 2009.**

Code(context)作为查询

- **Nguyen et al. ICSE 2012, Rahman et al. WCRE 2014.**

其它形式的查询

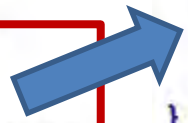
- **Stolee et al. TOSEM 2014, Inoue et al. ICSE 2012, Thummalapenta et al. ASE 2007, Holmes et al. ICSE 2005.**

自动代码修复

- **Zhong et al. ICSE 2015, Pei et al. ICSE 2015, Tao et al. FSE 2014, Kim et al. ICSE 2013.**

推荐粒度--代码片段

```
package net.micode.soundrecorder;
import android.media.AudioManager;
import android.media.MediaRecorder;
import android.telephony.TelephonyManager;
import java.io.File;
public class RecorderService {
  private void localStartRecording(){
    ...
    mRecorder = new MediaRecorder();
    mRecorder.setAudioSource(MediaRecorder.
    +AudioSource.MIC);
    ...
  }
  private void localStopRecording(){...}
  public static void startRecording(){...}
  public static void stopRecording(){...}
}
```



```
private void localStartRecording(){
  ...
  mRecorder = new MediaRecorder();
  mRecorder.setAudioSource(MediaRecorder.
  +AudioSource.MIC);
  ...
}
```

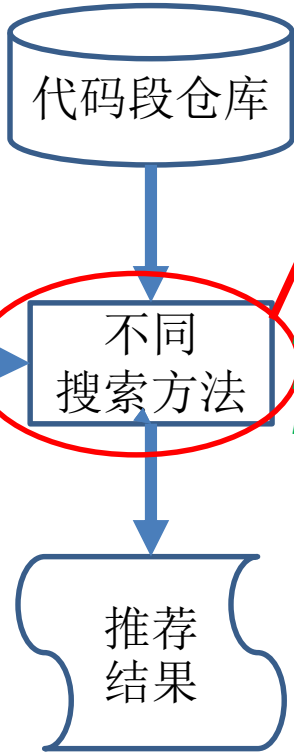
代码片段指的是Java 的一个类（class）中的某个方法。
它包含了注释和代码，代码有方法名和方法体。

面向任务的代码搜索

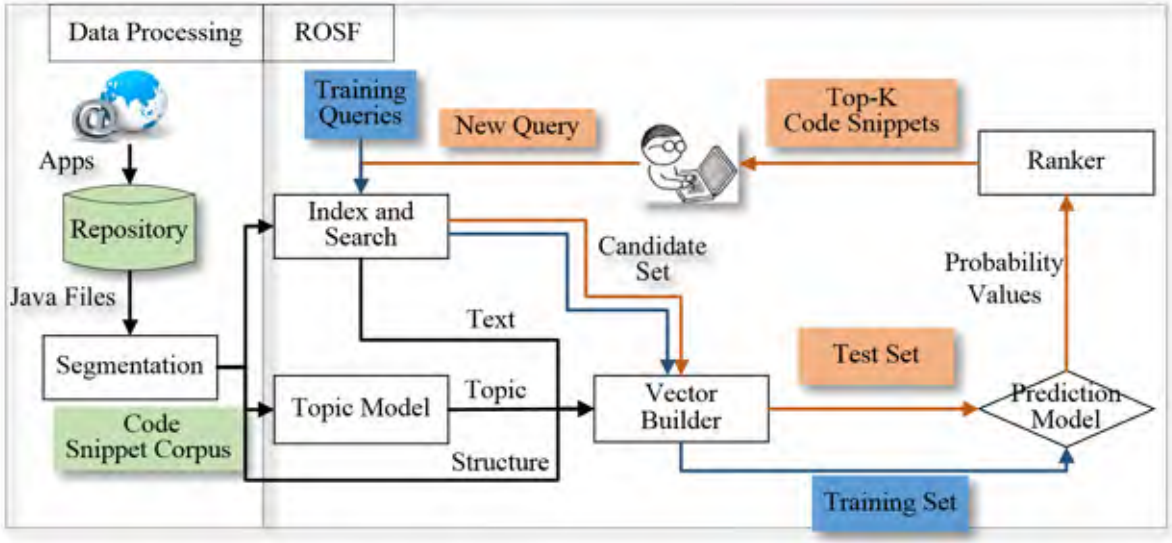
发现1：现有的代码推荐主要借助于信息检索的方法，匹配方式单一，效果不佳。

Solution：融合了信息检索和监督学习的方法。充分利用领域特征来构建分类器，为新查询推荐代码段。

自由文本的查询
record audio sound



自由文本作为查询的
代码推荐方法的一般框架



ROSF: Leveraging Information Retrieval and Supervised Learning for Recommending Code Snippets
（融合了信息检索和监督学习的代码推荐方法）

He Jiang*, Liming Nie, Zeyi Sun, Zhilei Ren, Weiqiang Kong, Tao Zhang, and Xiapu Luo,
“ ROSF: Leveraging Information Retrieval and Supervised Learning for Recommending Code Snippets”, *IEEE Transactions on Services Computing*,2016.

背景

- 当开发者输入**自由文本**的查询时，推荐方法如何为开发者提供一个**相关的**代码**片段列表**，辅助其完成特定的编程任务？

- 现有的代码推荐主要借助于**信息检索的方法**，匹配方式单一，有提升空间。如何**充分利用该任务中的多种领域特征**，并为它们自动分派不同的权重？

- 识别代码搜索中的**哪些特征**来构建分类器，进而为新的查询推荐代码片段？

方法--框架

提出**融合了信息检索和监督学习**的方法 (**ROSF**)。

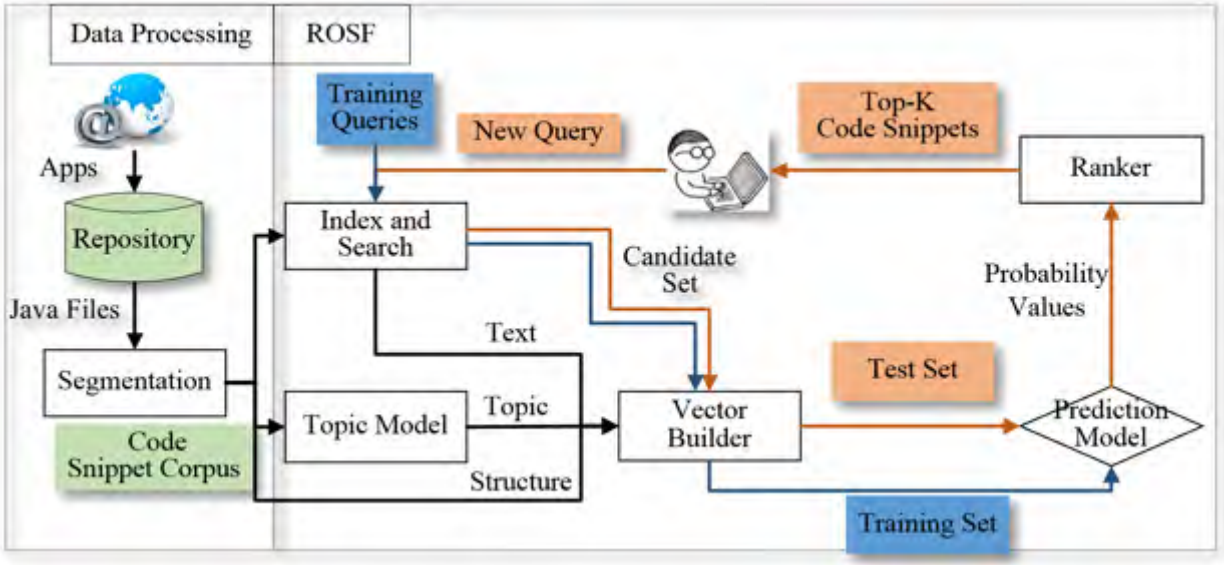
ROSF算法：

Step1：使用信息检索方法(BM25)为某些查询准备对应的代码段**候选集合**。

$$sim(D,q) = \sum_{t \in q \cap D} IDF(t) \cdot \frac{tf(t,D)(k_1+1)}{tf(t,D)+k_1\left(1-b+b\frac{|D|}{avgdl}\right)} \quad (1)$$

Step2：在识别一些**领域特征**并对候选集合中的代码片段进行**标注**的基础上，利用监督学习方法构建出预测模型。

Step3：当开发者输入新的查询时，预测模型为该查询对应的候选集合进行**重新排序**，最后推荐一定数目的代码段。



ROSF算法框架

蓝色带箭头的线表示训练的过程，
橘色带箭头的线表示推荐的过程，
黑色带箭头的线表示数据准备过程。

实验

- 查询

来自Stack Overflow中真实的编程任务，共35个。选择其中的20个做为测试集。

- 代码段仓库

921,713个代码片段，来自**1538**个开源app项目。

- 评估

4分值：高度相关（4），相关（3），一点相关（2），不相关（1）。

人工评分

实验

- 查询

QUERIES FOR TEST

ID	Query		ewed times
1	Record audio sound	android	83
2	Get screen dimensions in pixels	android, layout, scr	720031
3	Take a screenshot in Android	android, screenshot	107071
4	Get the memory used	android, memory, memory	217026
5	Get the list of activities/applications installed	android	180820
6	Import the system time	android, operating-system	36113
7	Open a URL in Android's web browser	android, url, android-intent, android-browser	342424
8	Use android Timer in Android activity	android, multithreading, timer, scheduled-tasks	18998
9	Capture Image from Camera and Display in Activity	android, image, camera, capture	157947
10	Handle right to left swipe gestures	android, swipe, gesture-recognition	152674
11	Converting pixels to dp	android	264672
12	Draw a line in android	android	182837
13	Get cpu usage	android, cpu-usage	72209
14	Detect network connection status	android, networking, wifi, connectivity	69553
15	Check if an application is installed or not in Android	android, apk	46174
16	Convert an image into Base64 string	android, base64	80049
17	Get the web page contents from a WebView	android, android-webwiew	52124
18	Cancel an executing AsyncTask	android, android-asynctask	85973
19	Detect if a Bluetooth device is connected	android	39245
20	Retrieve incoming call's phone number	android, telephonymanager, phone-state-listener	50134

代码段仓库等
数据已经公开！

实验设计--度量指标

- **Precision**
  - 一个查询的**推荐结果中相关代码片段数量所占的比例**。值越大越好。

$$\text{Precision} = \frac{|\text{Relevance}|}{|\text{Retrieved}|}$$

  - |Relevance| 表示推荐结果中相关代码片段的数量， |Retrieved| 表示推荐结果的数目。
  - 例如，为某个查询推荐10个代码片段，其中有7个是相关的，那么Precision的值为: 7/10*100% = 70%.

- **NDCG**
  - 根据代码片段在推荐列表中的位置来评估推荐方法的效果。**相关的代码片段被排的越靠前，NDCG值越大，性能越好。**
  - $\text{NDCG} = \frac{G}{iG}, \quad G = R_1 + \sum_{i=2}^{10} \frac{R_i}{log_2 i}$
  - R_1 和 R_i 分别表示出现在第1个位置和第i个位置上的代码片段对应的相关性得分。 iG表示推荐列表的理想排序。
- 使用两个度量指标在多个测试查询上的**平均值**作为算法性能的度量。
- 实验平台
  - PC配置如下：3.60 GHz CPU (Intel i5) 和 windows 8.1 操作系统.
  - 使用Java 编程语言，Eclipse开发环境。

实验设计--度量指标

- **Precision**
  - 一个查询的推荐结果中相关代码片段数量

$$\text{Precision} = \frac{|\text{Relevance}|}{|\text{Retrieved}|}$$

  - |Relevance| 表示推荐结果中相关代码片段的数量，|Retrieved| 表示推荐结果的数目。
  - 例如，为某个查询推荐10个代码片段，其中有7个是相关的，那么Precision的值为: 7/10*100% = 70%.

推荐结果中相关代码片段数量所占的比例，值越大越好。

- **NDCG**
  - 根据代码片段在推荐列表中的位置来[illegible]，NDCG值越大，性能越好。
  - $\text{NDCG} = \frac{G}{iG}$, $G = R_1 + \sum_{i=2}^{10} \frac{R_i}{log_2}$[illegible]
  - R_1 和 R_i 分别表示出现在第1个位置和第i个[illegible]关性得分。iG表示推荐列表的理想排序。

相关的代码片段被排的越靠前，NDCG值越大，性能越好。

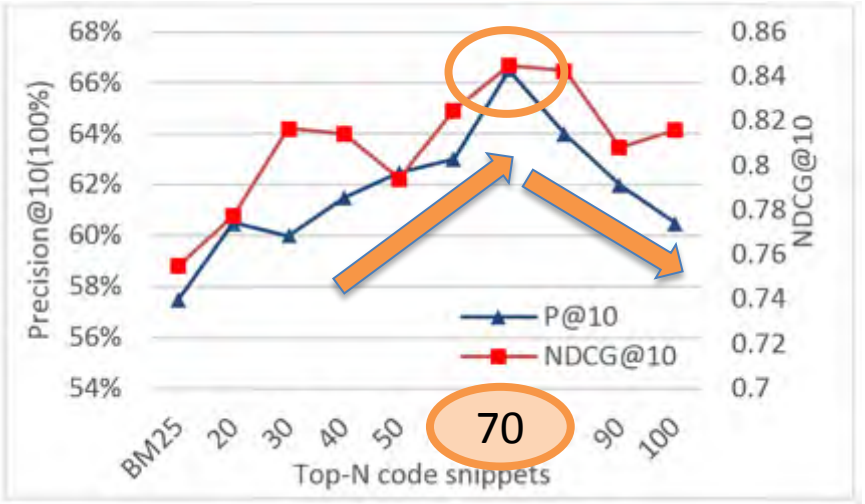
- 使用两个度量指标在多个测试查询上的平均值作为算法性能的度量。
- 实验平台
  - PC配置如下：3.60 GHz CPU (Intel i5) 和 windows 8.1 操作系统.
  - 使用Java 编程语言，Eclipse开发环境。

研究问题及结果-RQ1

RQ1: 是否参数（候选集合的规模）会影响算法的性能？

RECOMMENDATION PERFORMANCE OF ROSF WHEN N EQUALS TO DIFFERENT VALUES

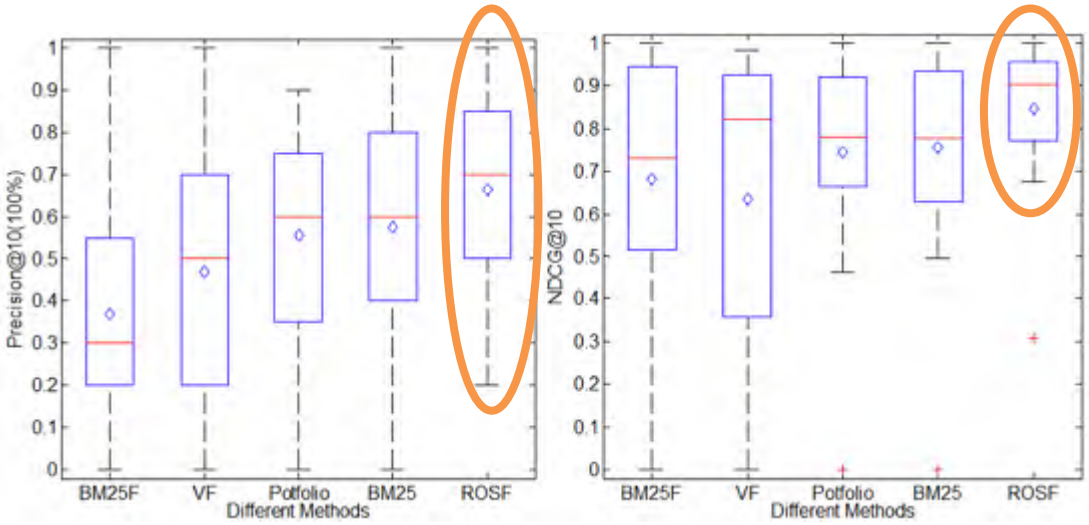
Top-N	Precision@10	NDCG@10
BM25	57.5%	0.7551
20	60.5%	0.7777
30	60%	0.8167
40	61.5%	0.8140
50	62.5%	0.7942
60	63%	0.8248
70	**66.5%**	**0.8448**
80	64%	0.8427
90	62%	0.8081
100	60.5%	0.8158



Answer to RQ1:当候选集合的规模递增时，算法性能（两个指标Precision@10 和NDCG@10）先增后降。当候选集合中实例数量达到70时，算法性能达到最好。

研究问题及结果-RQ2

RQ2: 与最新的代码推荐方法相比，本文方法(ROSF)效果如何？

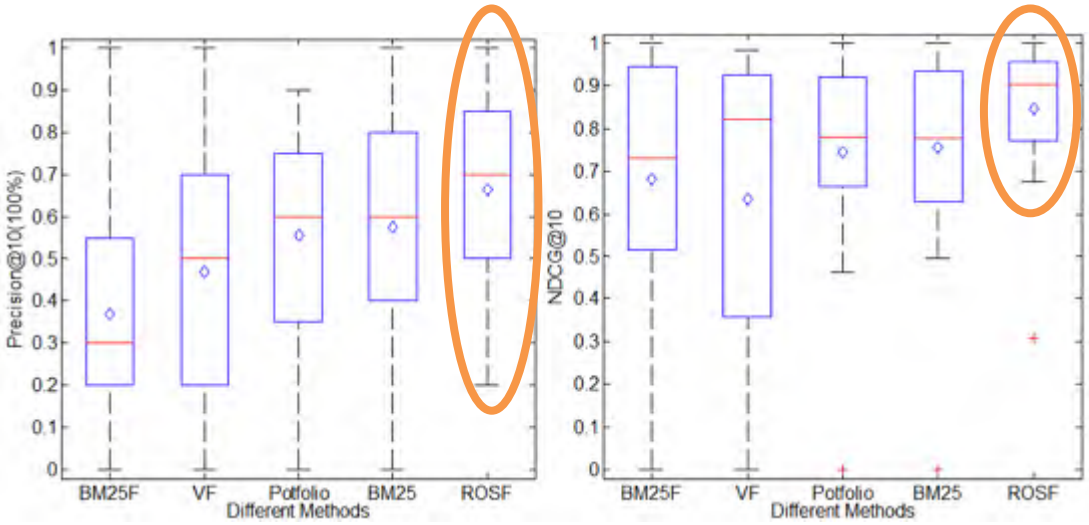


几种算法的结果对比

	Approach	Samples	Min	Max	Median	Mean	StdDev
Precision@10	ROSF	20	20%	100%	70%	66.5%	0.2390
	BM25	20	0%	100%	60%	57.5%	0.2552
	Portfolio	20	0%	90%	60%	55.5%	0.2350
	VF	20	0%	100%	50%	47%	0.3278
	BM25F	20	0%	100%	30%	37%	0.2755
NDCG@10	ROSF	20	0.3082	1	0.9041	0.8448	0.1646
	BM25	20	0	1	0.7772	0.7551	0.2407
	Portfolio	20	0	1	0.7795	0.7445	0.2325
	VF	20	0	0.9816	0.8220	0.6347	0.3526
	BM25F	20	0	1	0.7316	0.6819	0.2922

研究问题及结果-RQ2

RQ2: 与最新的代码推荐方法相比，本文方法(ROSF)效果如何？



几种算法的结果对比

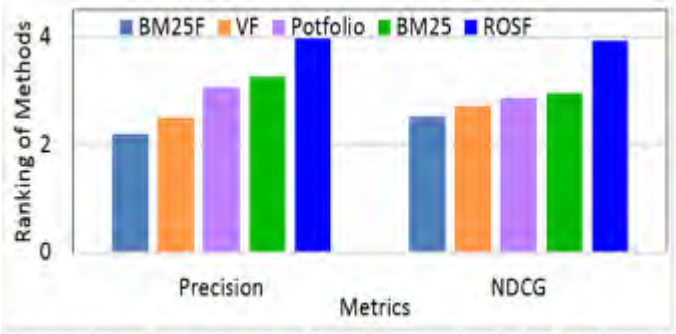
	Approach	Samples	Min	Max	Median	Mean	StdDev
Precision@10	ROSF	20	20%	100%	70%	66.5%	0.2390
	BM25	20	0%	100%	60%	57.5%	0.2552
	Portfolio	20	0%	90%	60%	55.5%	0.2350
	VF	20	0%	100%	50%	47%	0.3278
	BM25F	20	0%	100%	30%	37%	0.2755

Answer to RQ2:与几个最新的代码推荐方法对比，本文方法在Precision@10上提升了20%-41%，在NDCG@10上提升了13-33%

研究问题及结果-RQ2

RQ2: 与最新的代码推荐方法相比，本文方法(ROSF)效果如何？

为了得到**统计**意义上的结论，使用了Friedman's test 和two-sided Wilcoxon's signed rank tests



Comparisons for average ranking of five methods for Friedman's test

RESULTS OF WILCOXON'S TESTS BETWEEN ROSF AND OTHER COMPARATIVE METHODS FOR TWO METRICS

Metrics	R vs B	R vs P	R vs VF	R vs BF
Precision@10	0.0120	0.0059	0.0147	0.0019
NDCG@10	0.0000	0.0486	0.0057	0.0333

R refers to ROSF, B refers to BM25, P refers to Portfolio, and BF refers to BM25F. The first column indicates the two metrics. In other columns, the results of Wilcoxon's tests are reported for each metric. The comparison results consist of the p-value.

大部分P-value均小于0.05，说明存在统计意义上的区别。

对比算法：

TOSEM, 2013. Mcmillan et al. Portfolio Searching for Relevant Functions and Their Usages in Millions of Lines of Code.

ICSE, 2014. Keivanloo et al. Spotting Working Code Examples.

ICSE, 2010. Bajracharya et al. Leveraging usage similarity for effective retrieval of examples in code repositories.

研究问题及结果-RQ3

RQ3: 算法中使用到的几个特征是如何影响算法性能的？

首先,使用 Spearman‘s rank correlation coefficient (Spearman’s rho)分析了训练集中特征之间的相关性。

THE VALUES OF SPEARMAN'S RHO BETWEEN NINE FEATURES

	f1	f2	f3	f4	f5	f6	f7	f8	f9
f1	1	0.158	0.119	0.152	0.310	0.436	0.350	0.588	-0.041
f2		1	0.718	0.558	0.091	0.085	0.063	0.124	-0.151
f3			1	0.492	0.156	0.017	0.044	0.118	-0.132
f4				1	0.095	0.133	0.059	0.120	-0.113
f5					1	0.274	0.295	0.209	-0.008
f6						1	0.157	0.402	0.029
f7							1	0.274	0.016
f8								1	0.003
f9									1

特征相关性分析

可以看到，大部分特征之间的相关性都低于**0.6**，即特征之间的相关性较低，冗余较少。

研究问题及结果-RQ3

RQ3: 算法中使用到的几个特征是如何影响算法性能的？

特征对算法的影响

Ranking Methods	Precision@10		NDCG@10	
	Avg.	Impact	Avg.	Impact
ROSF	66.5%	-	0.844821	-
ROSF(except f1)	59.0%	-11.28%	0.833229	-1.37%
ROSF(except f2)	66.0%	-0.75%	0.847242	+0.29%
ROSF(except f3)	66.0%	-0.75%	0.848619	+0.45%
ROSF(except f4)	65.5%	-1.50%	0.852132	+0.87%
ROSF(except f5)	66.0%	-0.75%	0.848438	+0.43%
ROSF(except f6)	64.0%	-3.76%	0.868522	+2.81%
ROSF(except f7)	65.5%	-1.50%	0.845687	+0.10%
ROSF(except f8)	64.0%	-3.76%	0.830116	-1.74%
ROSF(except f9)	57.0%	-14.29%	0.757598	-10.32%

研究问题及结果-RQ3

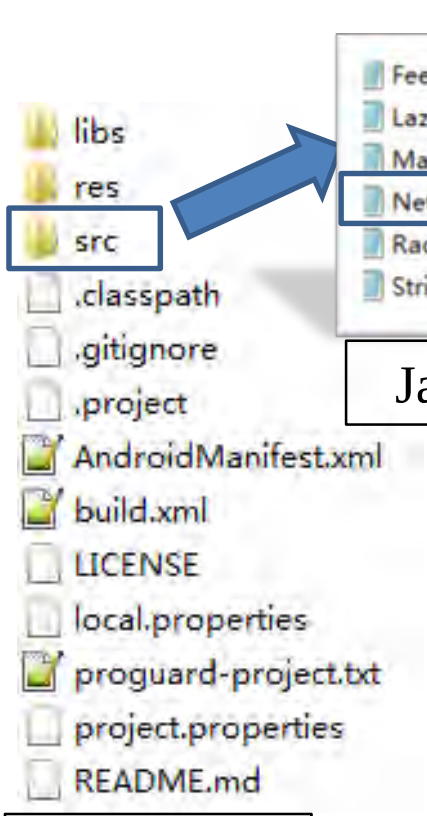
RQ3: 算法中使用到的几个特征是如何影响算法性能的？

特征对算法的影响

Ranking Methods	Precision@10		NDCG@10	
	Avg.	Impact	Avg.	Impact
ROSF	66.5%	-	0.844821	-
ROSF(except f1)	59.0%	-11.28%	0.833229	-1.37%
ROSF(except f2)	66.0%	-0.75%	0.847242	+0.29%
ROSF(except f3)	66.0%	-0.75%	0.848619	+0.45%
ROSF(except f7)	65.5%	-1.50%	0.845687	+0.10%
ROSF(except f8)	64.0%	-3.76%	0.830116	-1.74%
ROSF(except f9)	57.0%	-14.29%	0.757598	-10.32%

Answer to RQ3: 各个特征对算法影响程度各不相同。其中影响较大的几个特征分别是：代码段行数(f9)，查询与代码段内容的文本相似度(f1)，查询与Java引入语句之间的文本相似度(f6), 以及查询与代码段内容的主题相似度(f8)。

代码段仓库抽取流程



App 项目

Java 源代码

代码片段

命名

项目名 类名 方法名

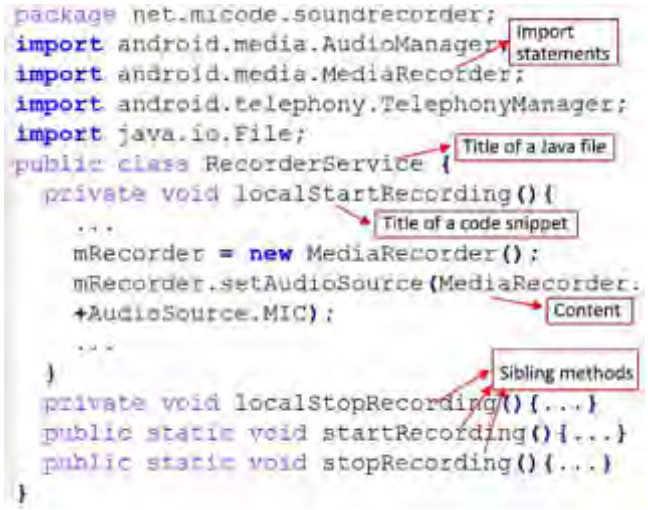
com.radiostudent.radiostudentstream_2_src.tar.gz@NetworkStateReceiver#onReceive.txt

代码段仓库

预处理 存储

代码段索引

三类领域特征



例子：Java 文件

文本相似度（与查询相关）

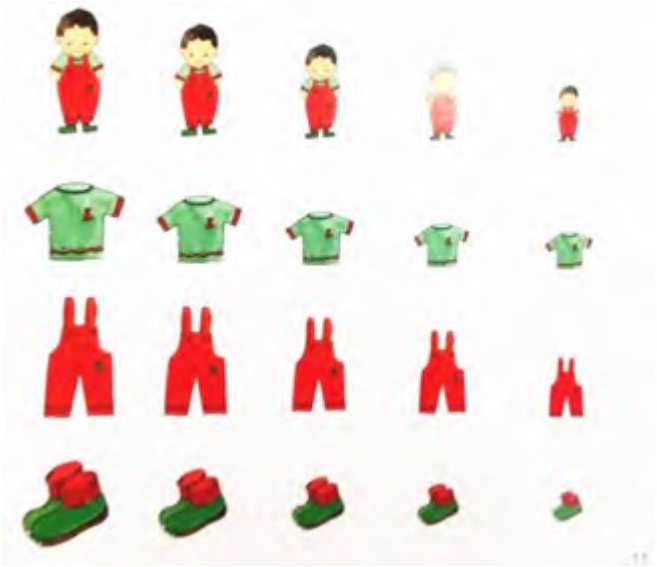
主题相似度（与查询相关）

结构特征（与查询无关）

ID	Categories	Features
f1		The textual similarity between a query and the content of a candidate code snippet (*i.e.*, code text and comments).
f2		The textual similarity between a query and the full title of a candidate code snippet (*i.e.*, package name of app, class name, and code snippet name).
f3		The textual similarity between a query and the simple title (only contains code snippet name) of a candidate code snippet.
f4	Text	The textual similarity between a query and the sibling method names contained in a same Java file with a candidate code snippet.
f5		The textual similarity between a query and the import statements of Android library in the Java file that contains a candidate code snippet.
f6		The textual similarity between a query and the import statements of the Java standard library in the Java file that contains a candidate code snippet.
f7		The textual similarity between a query and the import statements of other libraries in the Java file that contains a candidate code snippet.
f8	Topic	The topic similarity between a query and the content of a candidate code snippet.
f9	Structure	The number of lines in a candidate code snippet.

三类，9个特征

重排序



AN EXAMPLE FOR RE-RANKING THE CANDIDATE SET

	Relevance Scores				
Instances	1	2	3	4	Predicted Score
a	0.1	0.0	**0.9**	0.0	3
b	0.0	0.2	0.1	**0.7**	4
c	0.4	0.1	0.0	**0.5**	4
d	0.0	0.0	**0.6**	0.4	3
e	**0.8**	0.0	0.1	0.1	1

Code snippet with predicted score 4: b (0.7), c (0.5);
Code snippet with predicted score 3: a (0.9), d (0.6);
Code snippet with predicted score 2: null;
Code snippet with predicted score 1: e (0.8);
Top-3 recommendation results: b, c, and a.

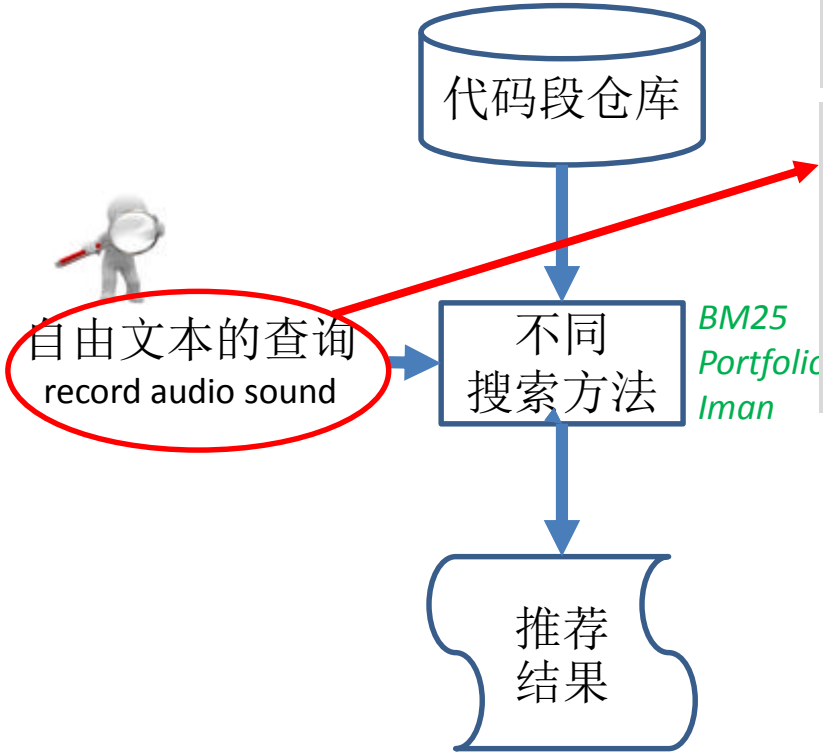
步骤：

Step1.为某个新查询对应侯选集合中的每个实例分别计算预测的4个分值的概率值。

Step2.根据每个分值中的概率值，为实例进行排序。

Step3.推荐N个代码片段给开发者。

面向任务的代码搜索

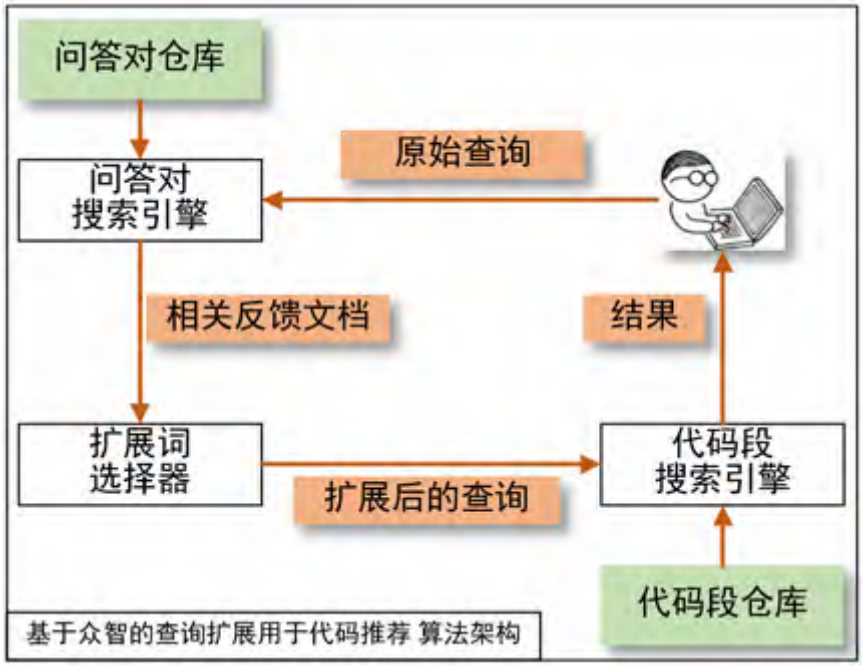


自由文本作为查询的
代码推荐方法的一般框架

发现1：现有的代码推荐主要借助于信息检索的方法，匹配方式单一，效果不佳。

发现2：查询词过短，以及查询和匹配内容使用不同的语言而导致词项失配问题

Solution：从Stack Overflow的问答对中抽取出编程相关的词来扩展原始查询。



QECK: Query Expansion Based on Crowd Knowledge for Code Search

（基于众智查询扩展的代码推荐）

Liming Nie, He Jiang*, Zhilei Ren, Zeyi Sun, Xiaochen Li, “Query Expansion Based on Crowd Knowledge for Code Search”, *IEEE Transactions on Services Computing*, 2016. PrePrints, doi:10.1109/TSC.2016.2560165.

背景

- 当开发者输入**自由文本**的查询时，推荐方法如何为开发者提供一个**相关的**代码**片段列表**，辅助其完成特定的编程任务？
- 由于**查询语句过短**，查询与代码段**使用不同的语言**，或者存在歧义造成检索效果不好而导致**词项失配问题**。如何解决？
- 如果查询扩展方法有用，那么如何找到**合适的扩展词**对原始查询进行扩展？

背景--相关工作的启发

- “Ranking crowd knowledge to assist software development,” Program Comprehension (PC), **2014**.
  把质量较高的 Stack Overflow(SO) **问答对** 推荐给开发者。

- D. Lo, "Query expansion via WordNet for effective code search," Software Analysis, Evolution and Reengineering (SANER), 2015.
  通过WordNet**对查询进行扩展**，进而推荐代码片段。

- D. Lo, “Automated construction of a software-specific word similarity database,” *Software Maintenance, Reengineering and Reverse Engineering (CSMRWCRE),* 2014.
- "SWordNet: Inferring semantically related words from software context,“ ESE, 2014.
  构建**软件相关的近义词**列表。

- 与本文工作同时期的其它工作：
- Lo D. RACK: Automatic API Recommendation using Crowdsourced Knowledge, SANER, 2016.
  介绍了使用众智进行API推荐

背景--相关工作的启发

- “Ranking crowd knowledge to assist software development,” Program Comprehension (PC), **2014**.
  把质量较高的 Stack Overflow(SO) **问答对** 推荐给开发者。

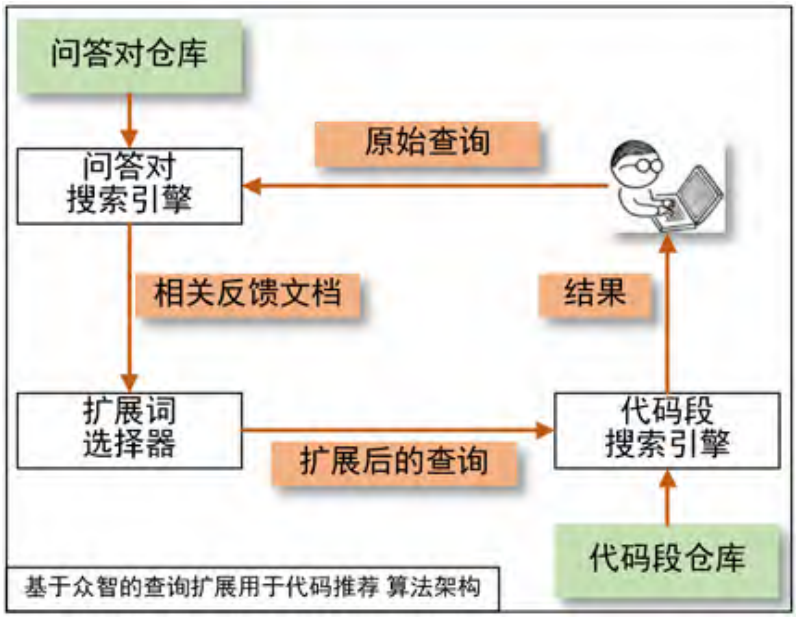
- D. Lo, "Query expansion via WordNet for effective code search," Software Analysis, Evolution and Reengineering (SANER), 2015.
  通过WordNet**对查询进行扩展**，进而推荐代码片段。

思考：是否可以从Stack Overflow中抽取软件相关的扩展词用于查询的预操作，进而推荐更好的代码片段？

- 与本文工作同时期的其它工作：
- Lo D. RACK: Automatic API Recommendation using Crowdsourced Knowledge, SANER, 2016.
  介绍了使用众智进行API推荐

方法(QECK)

利用伪相关反馈方法从Stack Overflow的问答对(众智)中抽取出扩展词，对原始查询进行扩展。



基于众智查询扩展的代码推荐算法框架

步骤：

1. 利用文本相似度和用户评分的**综合得分**来推荐与原始查询最相关的问答对；

$$final\ score_i = \frac{L_i - minL}{maxL - minL} + \frac{S_i - minS}{maxS - minS} \quad (1)$$

$$S_i = 0.7 * Sq_i + 0.3 * Sa_i \quad (2)$$

2. 利用伪相关反馈方法从问答对中选出与原始查询相关的扩展词；

具体来说，为反馈文档中的每个词设定一个权重。在本文档中出现的次数多，在别的文档中出现的次数少，则该词的权重值越大。

$$w(t, d_i) = TF(t) * IDF(t) \quad (3)$$

$$TF(t, d_i) = sqrt(tf(t, d_i)) \quad (4)$$

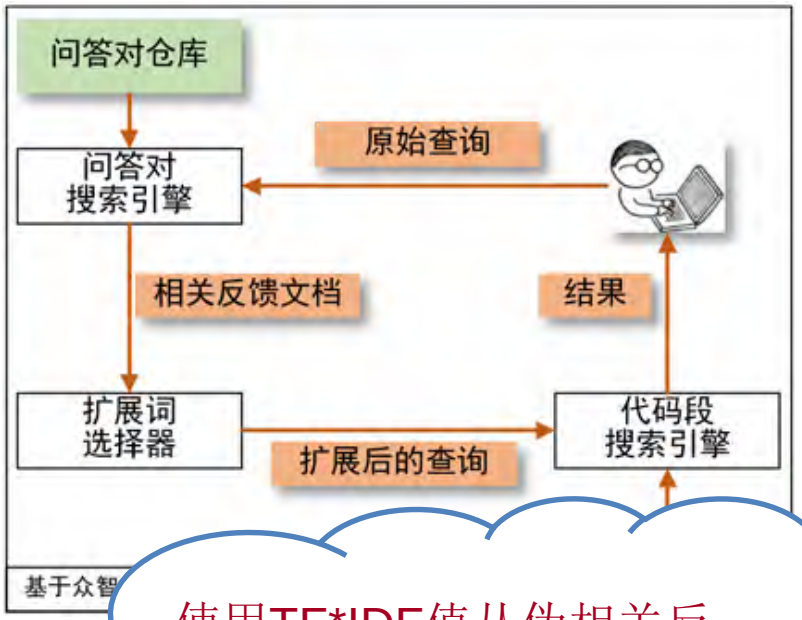
$$IDF(t) = log\left(\frac{N}{df+1}\right) + 1 \quad (5)$$

3. 把获取的扩展词加入到原始查询中，利用扩展后的查询对代码段仓库进行检索。

方法(QECK)

利用伪相关反馈方法从Stack Overflow词，对原始查询进行扩展。

使用了两种得分的平均值来获取伪相关反馈文档：文本相似度和用户评分。



步骤：

1. 利用文本相似度和用户评分的**综合得分**来推荐与原始查询最相关的问答对；

$$final\ score_i = \frac{L_i - minL}{maxL - minL} + \frac{S_i - minS}{maxS - minS} \quad (1)$$

$$S_i = 0.7 * Sq_i + 0.3 * Sa_i \quad (2)$$

2. 利用伪相关反馈方法从问答对中选出与原始查询相关的扩展词；

具体来说，为反馈文档中的每个词设定一个权重。在本文档中出现的次数多，在别的文档中出现的次数少，则该词的权重值越大。

$$w(t, d_i) = TF(t) * IDF(t) \quad (3)$$

$$TF(t, d_i) = sqrt(tf(t, d_i)) \quad (4)$$

$$IDF(t) = log\left(\frac{N}{df+1}\right) + 1 \quad (5)$$

使用TF*IDF值从伪相关反馈文档的词集合中抽取有代表性的词。

3. 把获取的扩展词加入到原始查询中，利用扩展后的查询对代码段仓库进行检索。

实验

- **查询**

  20个真实的编程任务，来自Stack Overflow。

- SO问答对仓库

  24,120,522 帖子--- 5,108,770 问答对---312,941Android 问答对。

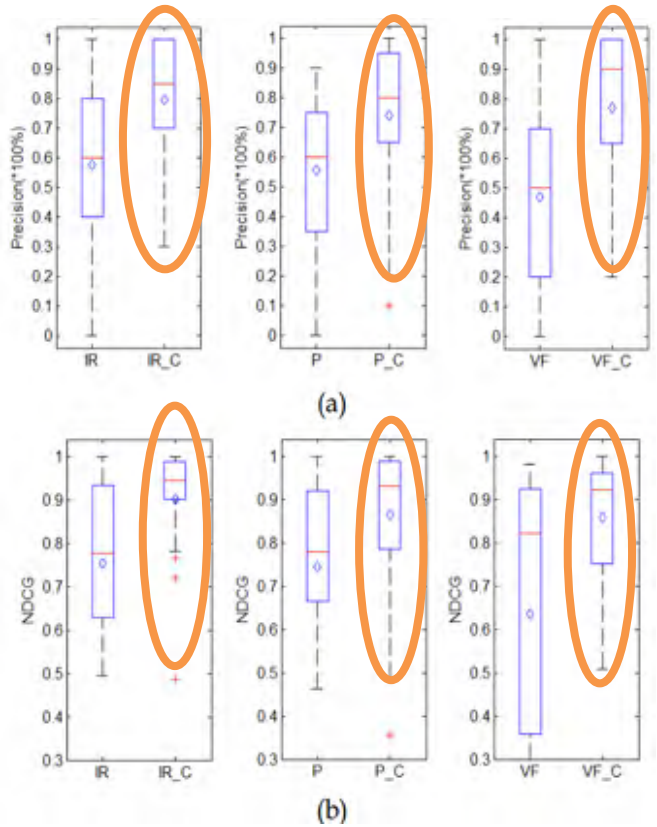
  （屏蔽掉与查询相同的问答对）

- 代码段仓库

  921,713个代码片段，来自1538个app项目。

研究问题及结果-RQ1

RQ1: QECK是否能够提升现有代码搜索算法的性能?



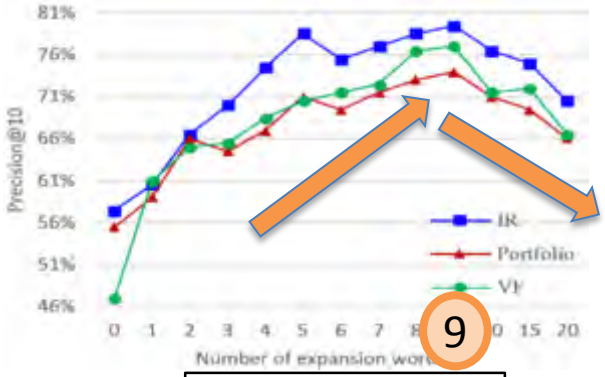
本方法在三个算法上都取得了明显的提升效果，
在Precision@10上分别提升了：
38%，33%，64%
在NDCG@10上分别提升了：
20%，16%，35 %

对比算法：
IR: BM25
P: Portfolio (TOSEM 2013)
VF: Iman (ICSE 2014)

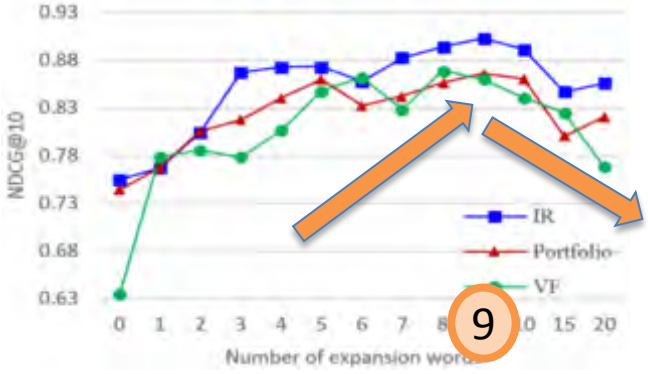
对比算法：
TOSEM, 2013. Portfolio Searching for Relevant Functions and Their Usages in Millions of Lines of Code.
ICSE, 2014. Spotting Working Code Examples.

研究问题及结果-RQ2

RQ2: 参数是如何影响推荐性能的？

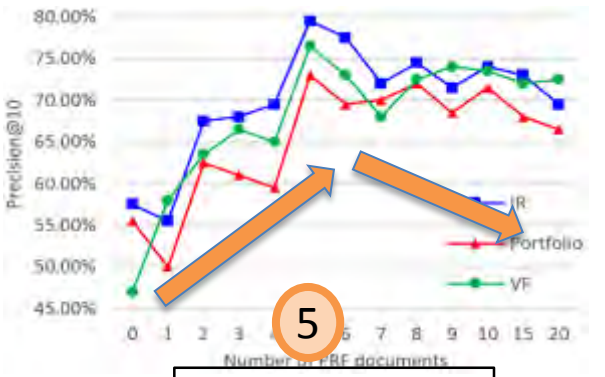


Precision

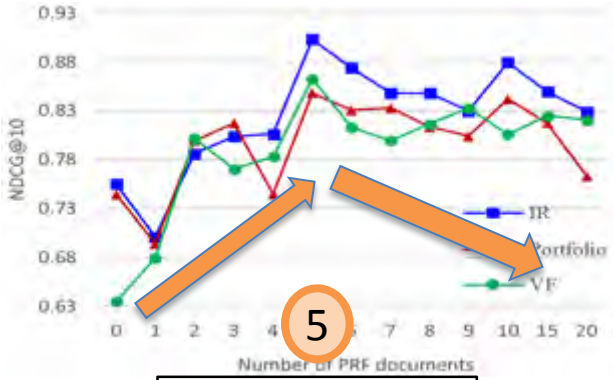


NDCG

伪相关反馈文档数目固定为5，扩展词个数变化时对应两个度量指标的变化情况



Precision

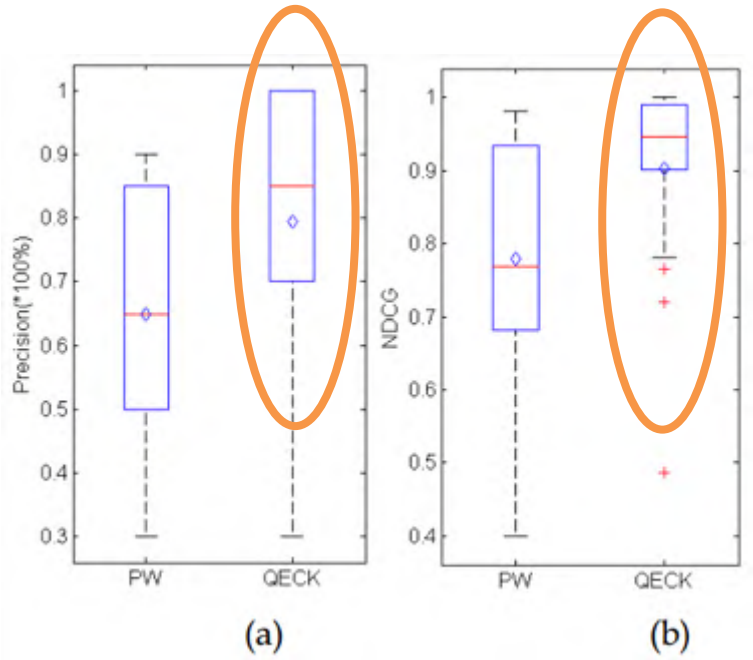


NDCG

扩展词个数固定为9，伪相关反馈文档数目变化时对应两个指标的变化情况

研究问题及结果-RQ3

RQ3: 与最新的查询扩展方法相比，本文方法效果如何？



PW: P-WordNet (SANER 2015)

与最新的查询扩展代码推荐方法对比
在Precision@10上提升了22%
在NDCG@10上提升了16%

对比算法：
SANER, 2015, D. Lo, Query expansion via WordNet for effective code search

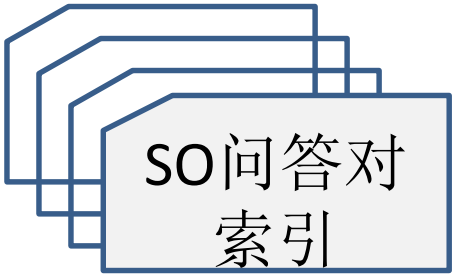
实验设计--SO问答对仓库

一个SO问答对的处理过程

过滤
- 通过Tag过滤，得到Android编程相关的问答对

预处理
- 对问答对进行文本预处理，去停用词，驼峰分词，取词干等操作

存储
- 每个问答对作为一个document对象保存在**Lucene**中。一个document有多个域
- 便于快速检索



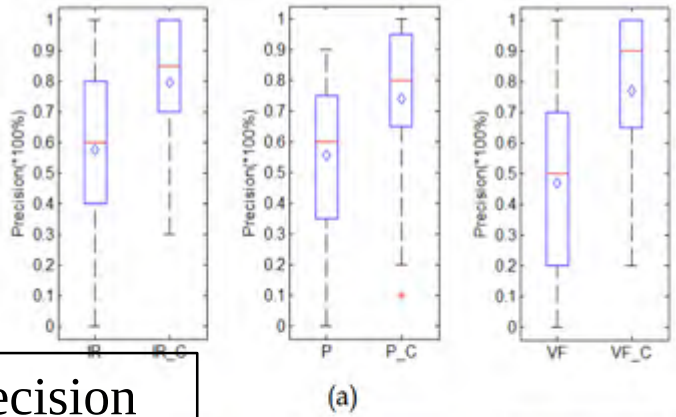
Q: programmat screenshot android screenshot select area phone screen program code

A: code allow screenshot store sd card add proper permiss save file permiss android android permiss write extern storag code run activ ...

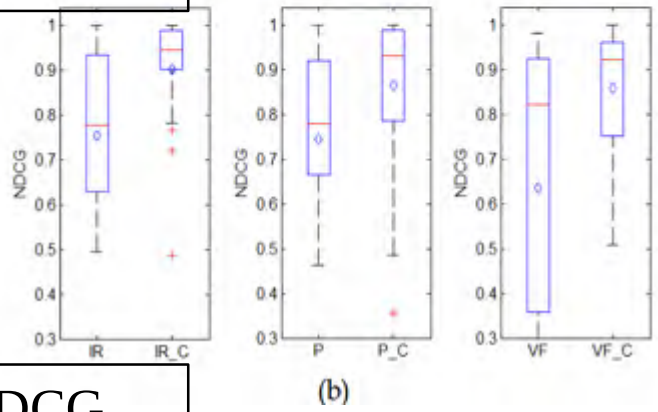
Q&A pair ID	words
2661536	programmat screenshot android screenshot select area phone screen program code code allow screenshot store sd card add proper permiss save file permiss...
...	...

Lucene是一个开源的信息检索工具。它能提供快速的建索引及搜索的功能。

QECK VS ROSF

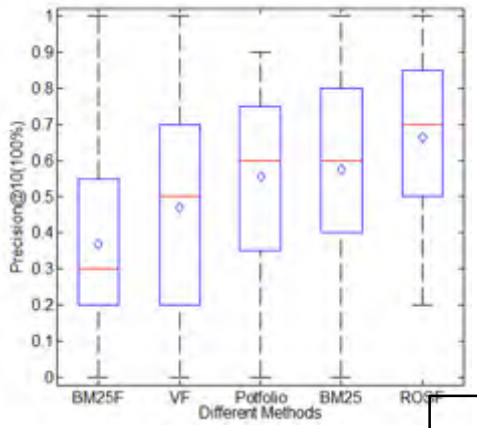


Precision

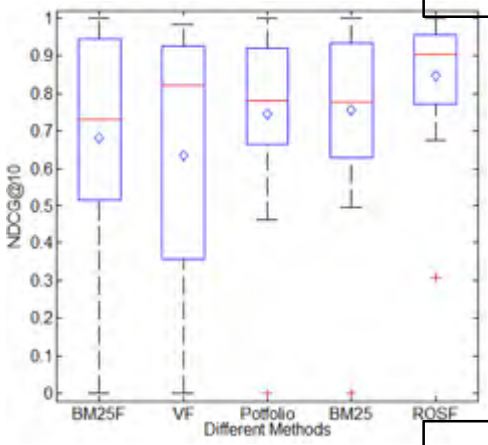


NDCG

VS



Precision



NDCG

QECK

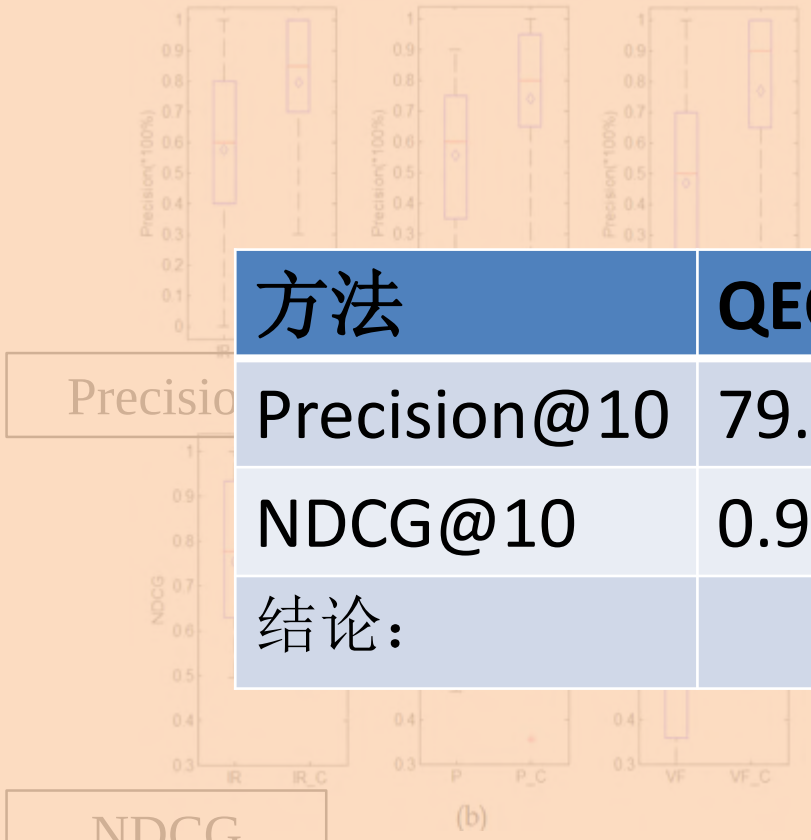
基于众智查询扩展

ROSF

融合了信息检索和监督学习

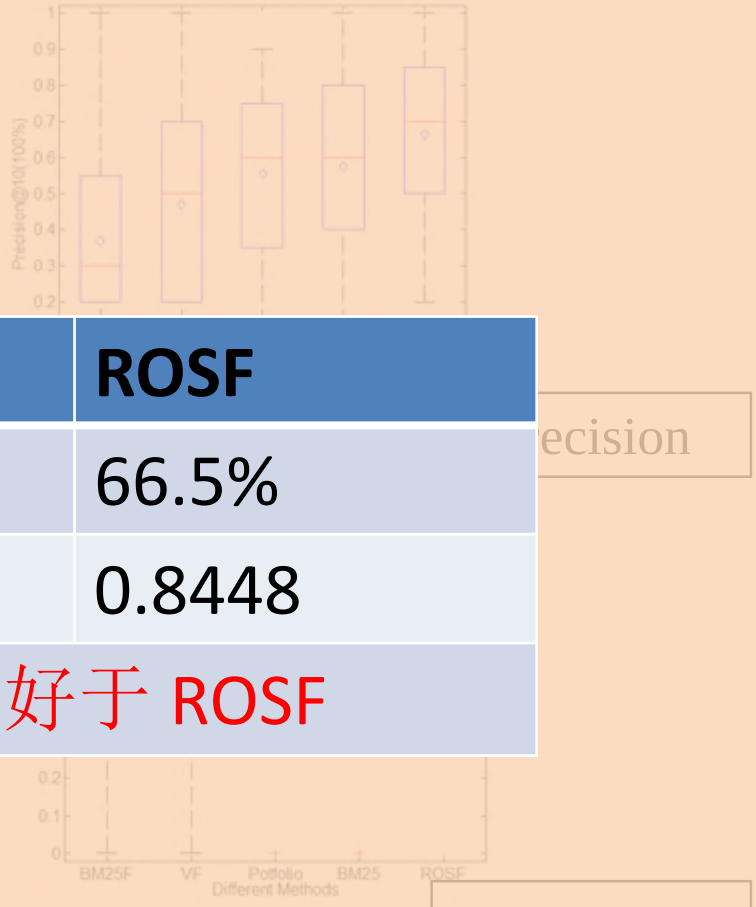
QECK VS ROSF

方法	QECK	ROSF
Precision@10	79.5%	66.5%
NDCG@10	0.9030	0.8448
结论：	QECK 好于 ROSF	



QECK

基于众智查询扩展



ROSF

融合了信息检索和监督学习

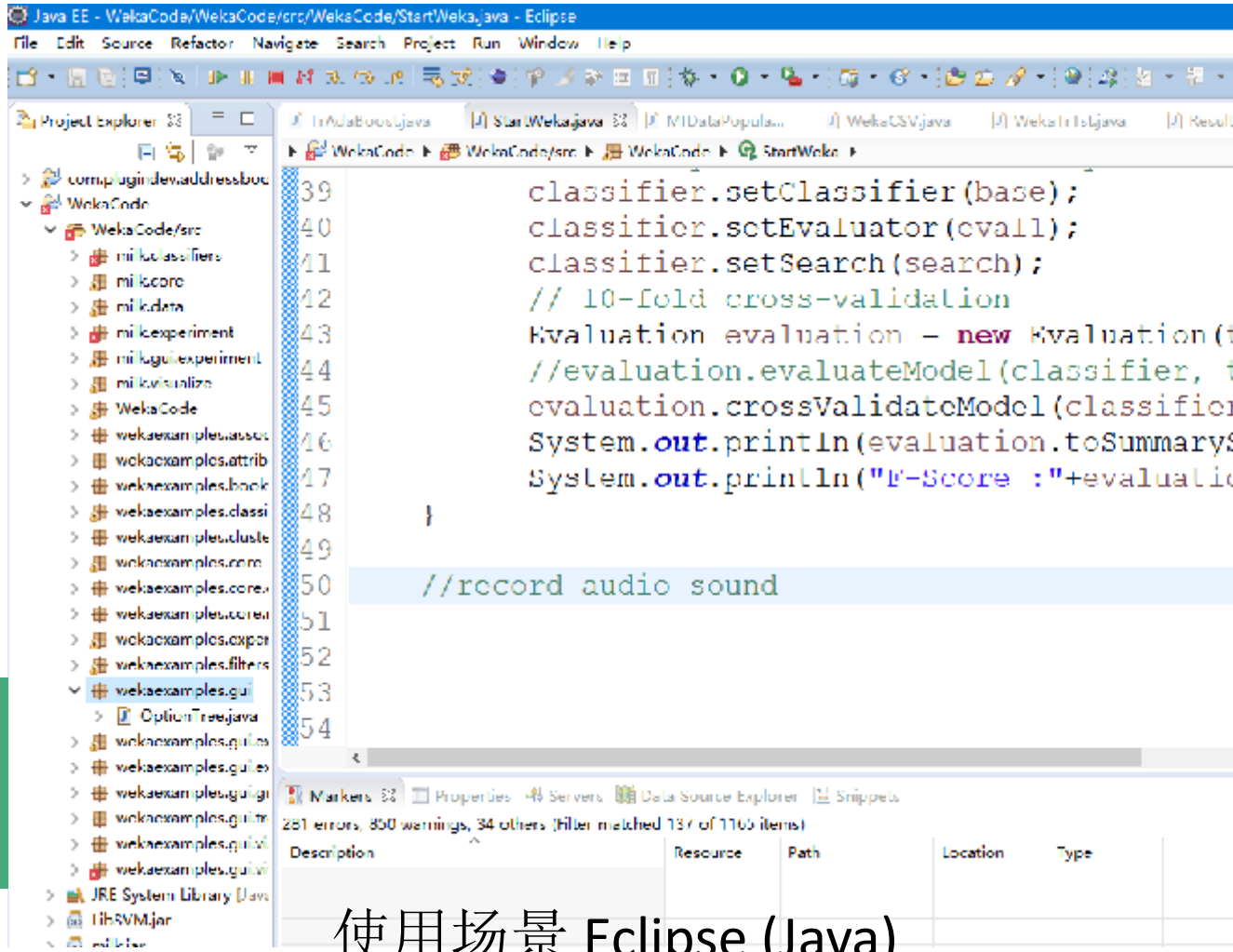
Eclipse 嵌入式工具

使用步骤：

Step1.选中需要搜索的查询文本，点击鼠标右键，找到“代码搜索”选项并点击。在Eclipse界面下方会呈现10个推荐的相关代码段列表。(也可使用快捷键Ctrl+Shift+V)

Step2.双击结果中的某个代码片段，在编程界面会呈现该代码段的内容供开发者参考。

平均每次搜索过程大概1-2秒。这得益于Lucene平台的使用。Lucene能提供高效的存储和检索服务。



使用场景 Eclipse (Java)

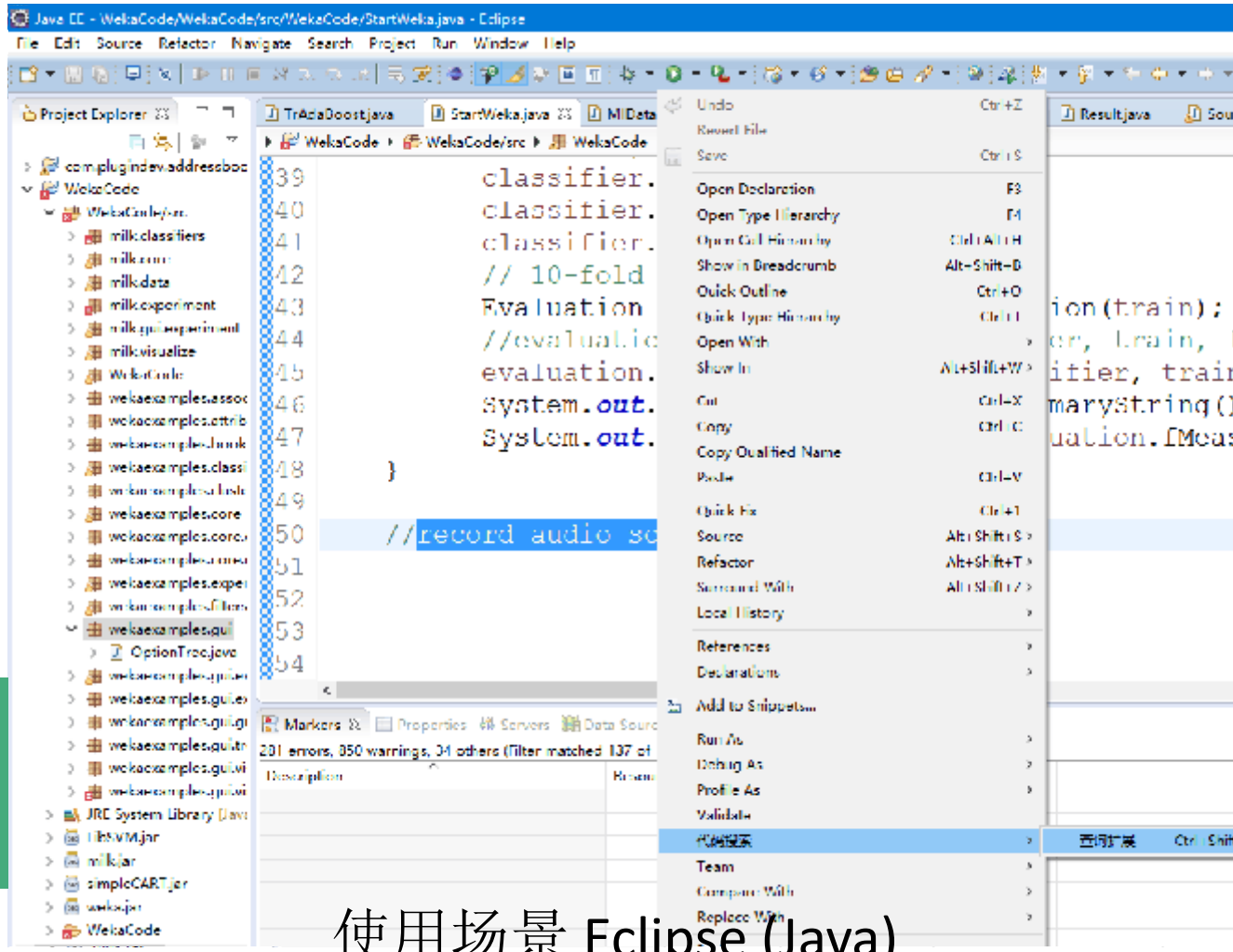
Eclipse 嵌入式工具

使用步骤：

Step1.选中需要搜索的查询文本，点击鼠标右键，找到“代码搜索”选项并点击。在Eclipse界面下方会呈现10个推荐的相关代码段列表。(也可使用快捷键Ctrl+Shift+V)

Step2.双击结果中的某个代码片段，在编程界面会呈现该代码段的内容供开发者参考。

平均每次搜索过程大概1-2秒。这得益于Lucene平台的使用。Lucene能提供高效的存储和检索服务。



使用场景 Eclipse (Java)

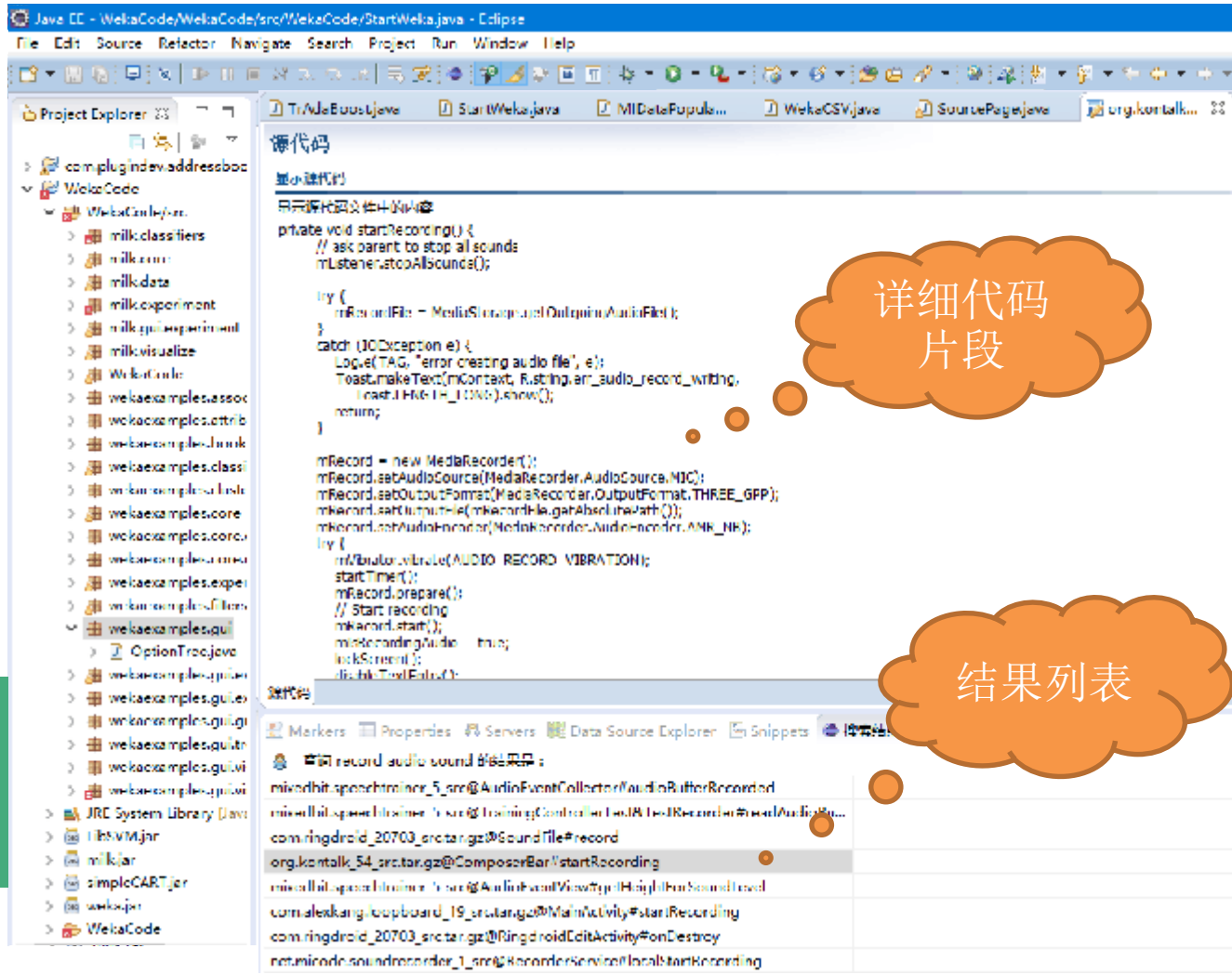
Eclipse 嵌入式工具

使用步骤：

Step1.选中需要搜索的查询文本，点击鼠标右键，找到“代码搜索”选项并点击。在Eclipse界面下方会呈现10个推荐的相关代码段列表。(也可使用快捷键Ctrl+Shift+V)

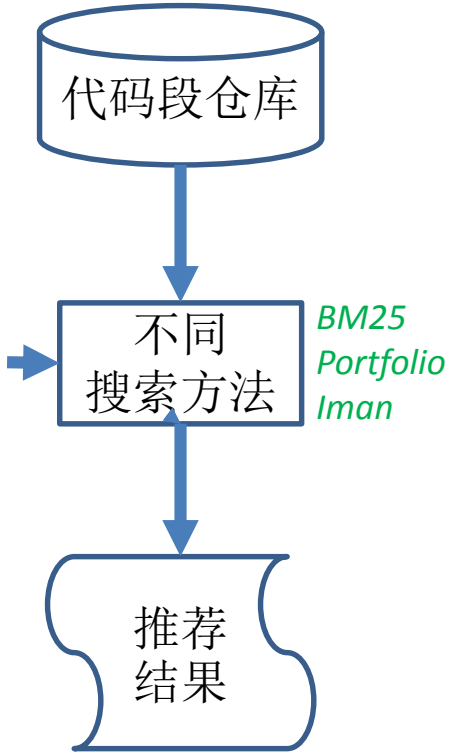
Step2.双击结果中的某个代码片段，在编程界面会呈现该代码段的内容供开发者参考。

平均每次搜索过程大概1-2秒。这得益于Lucene平台的使用。Lucene能提供高效的存储和检索服务。



总 结

自由文本的查询
record audio sound



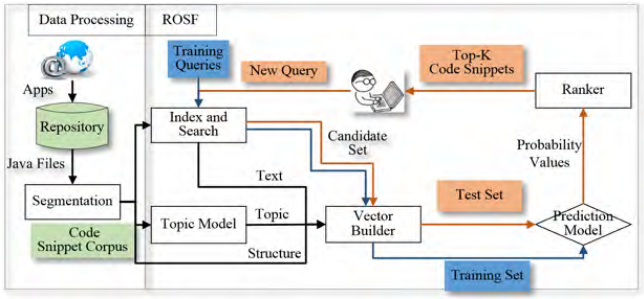
自由文本作为查询的
代码推荐方法的一般框架

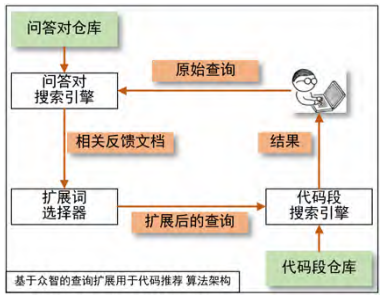
发现1：现有的代码推荐主要借助于信息检索的方法，匹配方式单一，效果不佳。

Solution：融合了信息检索和监督学习的方法。充分利用领域特征来构建分类器，为新查询推荐代码段。

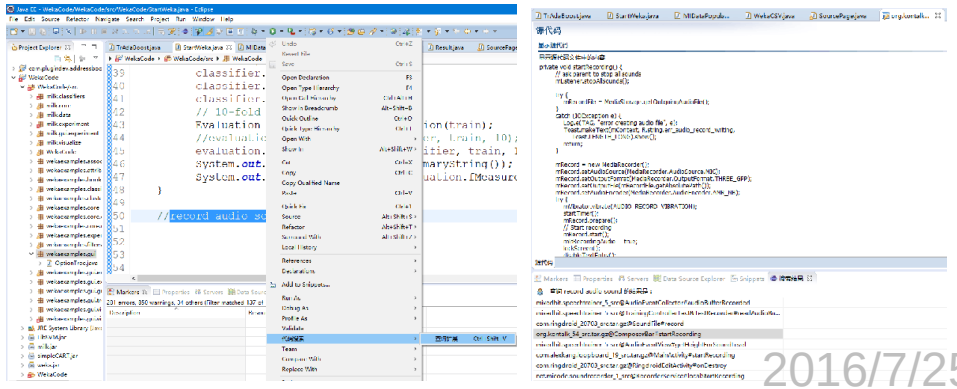
发现2：查询词过短，以及查询和匹配内容使用不同的语言而导致词项失配问题

Solution：从Stack Overflow的问答对中抽取出编程相关的词来扩展原始查询。





Eclipse 插件



Thanks!