

Statistical-based Clone Detection

He Feng

Department of Physics

fenghe@vt.edu

Liuqing Li

Department of Computer Science

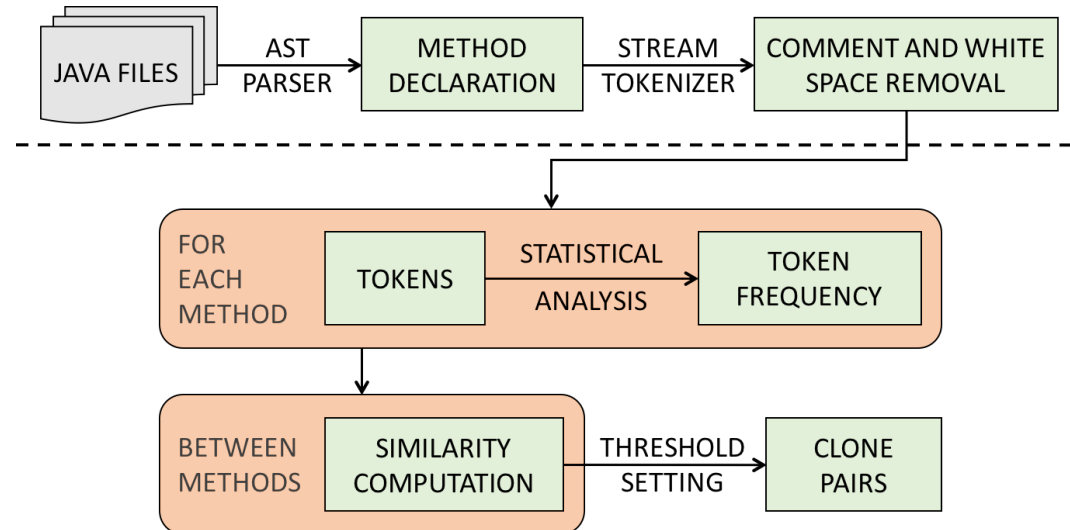
liuqing@vt.edu

Outline

- Problem and Solution
- Technical Challenges
- Experiment Results
- Things We Learn

Problem and Solution

- Problem and Goal
 - Code Clone Detection is important
 - To detect code clones between methods based on tokens
- Solution: STCD
 - Overall Prior Project Diagram



Technical Challenges

- Challenges in mid-term presentation
 - ASTParser
 - Variables Comparison
 - Similarity Calculation
 - Manual Weights
 - Data Collection
 - Results Comparison
 - UI Development

X

X

X

X

X

X

X

Technical Challenges

- Challenges in mid-term presentation
 - ASTParser
 - Variables Comparison
 - Similarity Calculation
 - Manual Weights
 - Data Collection
 - Results Comparison
 - UI Development

X

X

X

X

X

X

X

Technical Challenges

- ASTParser



- Excessive time cost
- Improve the method parsing process
- e.g. Regular expression
- Build-in functions are helpful

- Variables Comparison



- e.g. errorLineMessage and messageErrorLine
- Bigram Algorithm has been applied
- Threshold is 0.7

Technical Challenges

- Challenges in mid-term presentation
 - ASTParser
 - Variables Comparison
 - Similarity Calculation
 - Manual Weights
 - Data Collection
 - Results Comparison
 - UI Development

X

✓

X

X

X

X

X

Technical Challenges

- Similarity Calculation
 - Old Similarity Algorithm for token Lists
 - $\text{Sim}(\text{List}_x, \text{List}_y) = 1 / (1 + \text{Distance}(\text{List}_x, \text{List}_y))$
 - $\text{Distance}(\text{List}_x, \text{List}_y)$: Euclidean Distance
 - Not consider the list length
 - Example 1
 - $\text{List}_x = \{ \langle a, 3 \rangle, \langle b, 3 \rangle, \dots, \langle y, 3 \rangle, \langle z, 1 \rangle \}$
 - $\text{List}_y = \{ \langle a, 3 \rangle, \langle b, 3 \rangle, \dots, \langle y, 3 \rangle, \langle z, 10 \rangle \}$
 - $\text{Sim}(\text{List}_x, \text{List}_y) = 1 / (1 + 9) = 0.1$
 - Example 2
 - $\text{List}_x = \{ \langle a, 3 \rangle, \langle b, 5 \rangle, \langle c, 4 \rangle, \langle d, 1 \rangle \}$
 - $\text{List}_y = \{ \langle a, 0 \rangle, \langle b, 2 \rangle, \langle c, 1 \rangle, \langle d, 4 \rangle \}$
 - $\text{Sim}(\text{List}_x, \text{List}_y) = 1 / (1 + 6) = 0.14$

Vector
methodPara
methodType
tokenListNum
tokenListType
tokenListKeyword
tokenListMarker
tokenListOperator
tokenListOther1
tokenListOther2

Technical Challenges

- Similarity Calculation



- New Similarity Algorithm for token Lists

- $\text{Diff}(\text{Elem}_{x_a}, \text{Elem}_{y_a}) = \text{abs}(\text{Freq}_{x_a} - \text{Freq}_{y_a})$
 - $\text{Sim}(\text{List}_x, \text{List}_y) = 1 - \sum \text{Diff}(\text{Elem}_{x_i}, \text{Elem}_{y_i}) / (\sum \text{Freq}_{x_i} + \sum \text{Freq}_{y_i})$
 - Similar to Levenshtein distance

- Example 1

- $\text{List}_x = \{ \langle a, 3 \rangle, \langle b, 3 \rangle, \dots, \langle y, 3 \rangle, \langle z, 1 \rangle \}$
 - $\text{List}_y = \{ \langle a, 3 \rangle, \langle b, 3 \rangle, \dots, \langle y, 3 \rangle, \langle z, 10 \rangle \}$
 - $\text{Sim}(\text{List}_x, \text{List}_y) = 1 - 9 / (76 + 85) = 0.94$

- Example 2

- $\text{List}_x = \{ \langle a, 3 \rangle, \langle b, 5 \rangle, \langle c, 4 \rangle, \langle d, 1 \rangle \}$
 - $\text{List}_y = \{ \langle a, 0 \rangle, \langle b, 2 \rangle, \langle c, 1 \rangle, \langle d, 4 \rangle \}$
 - $\text{Sim}(\text{List}_x, \text{List}_y) = 1 - 12 / (13 + 7) = 0.4$

Vector
methodPara
methodType
tokenListNum
tokenListType
tokenListKeyword
tokenListMarker
tokenListOperator
tokenListOther1
tokenListOther2

Technical Challenges

- Challenges in mid-term presentation
 - ASTParser
 - Variables Comparison
 - Similarity Calculation
 - **Manual Weights**
 - Data Collection
 - Results Comparison
 - UI Development

X

✓

✓

X

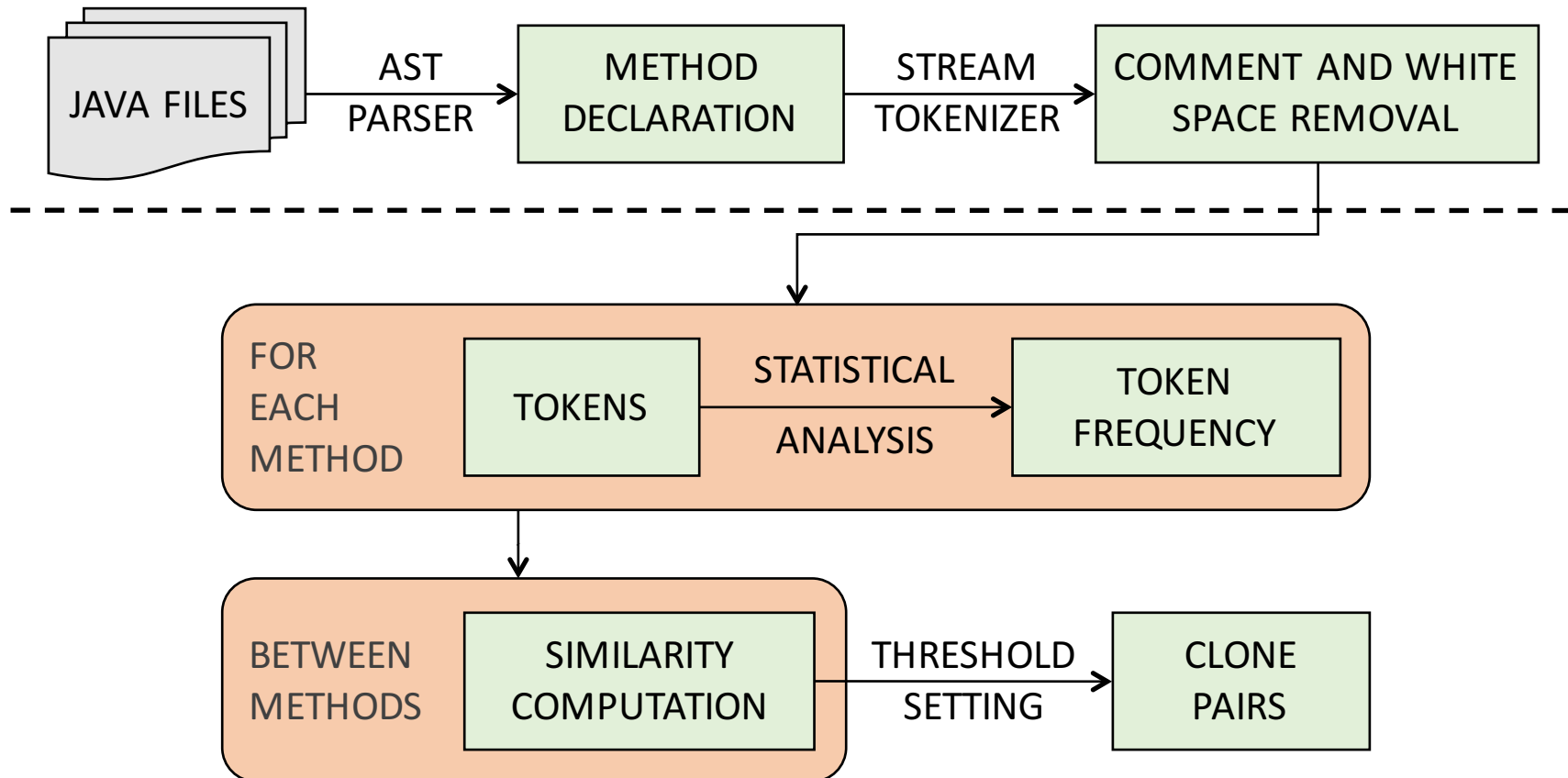
X

X

X

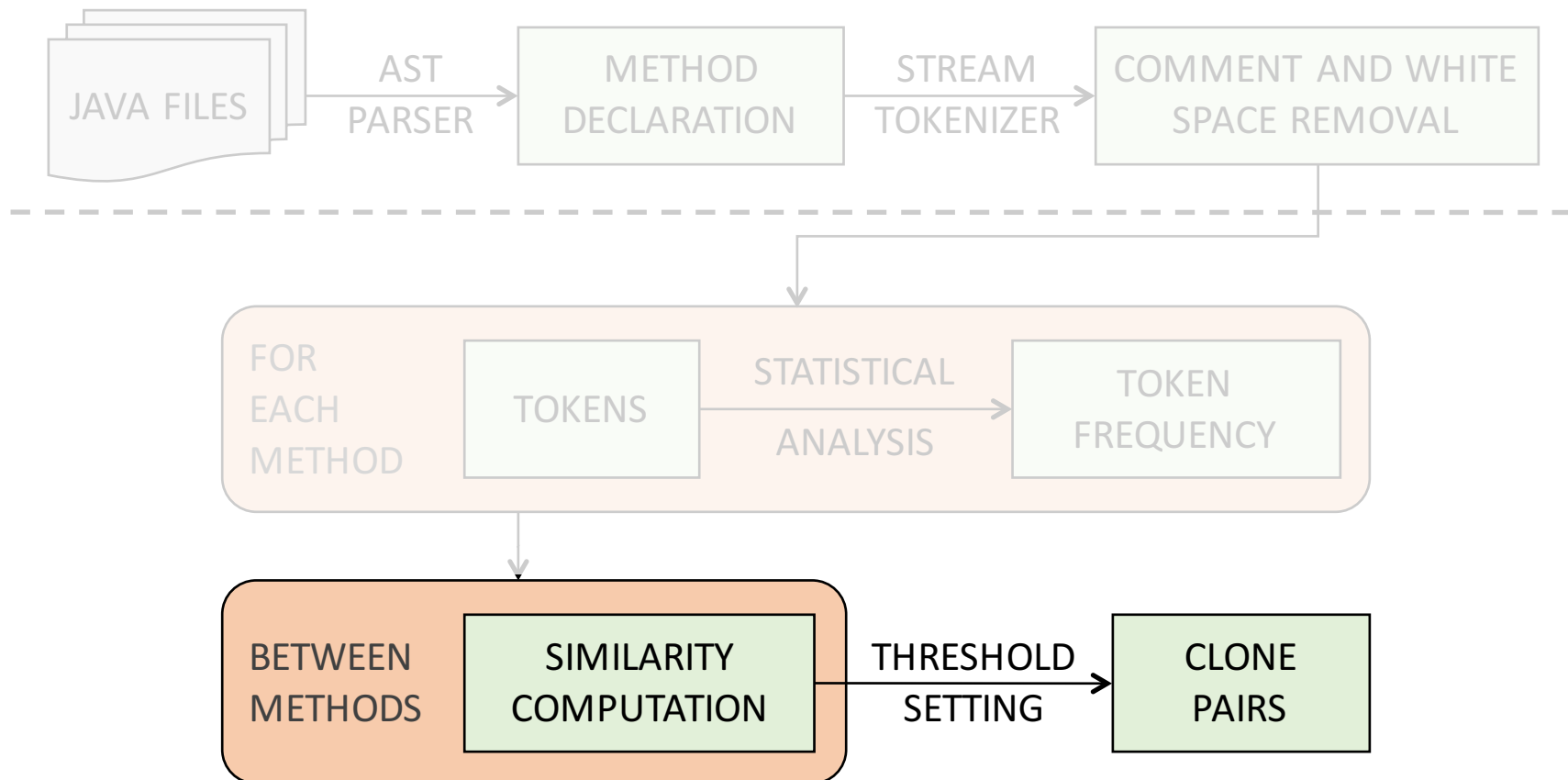
Technical Challenges

- Manual Weights



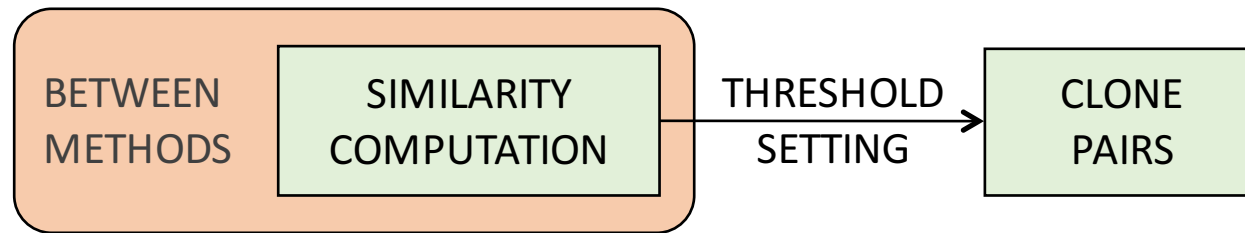
Technical Challenges

- Manual Weights

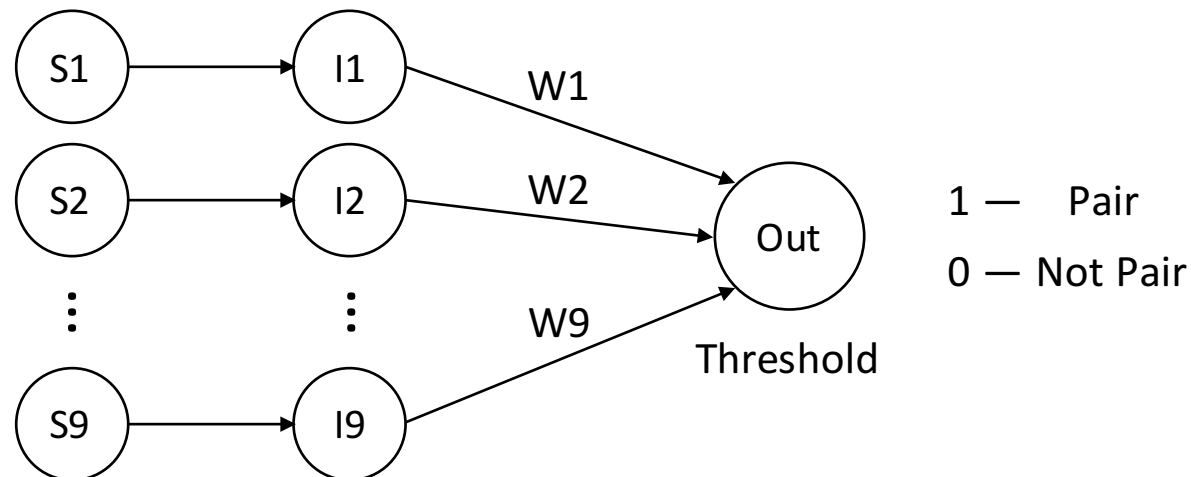


Technical Challenges

- Manual Weights

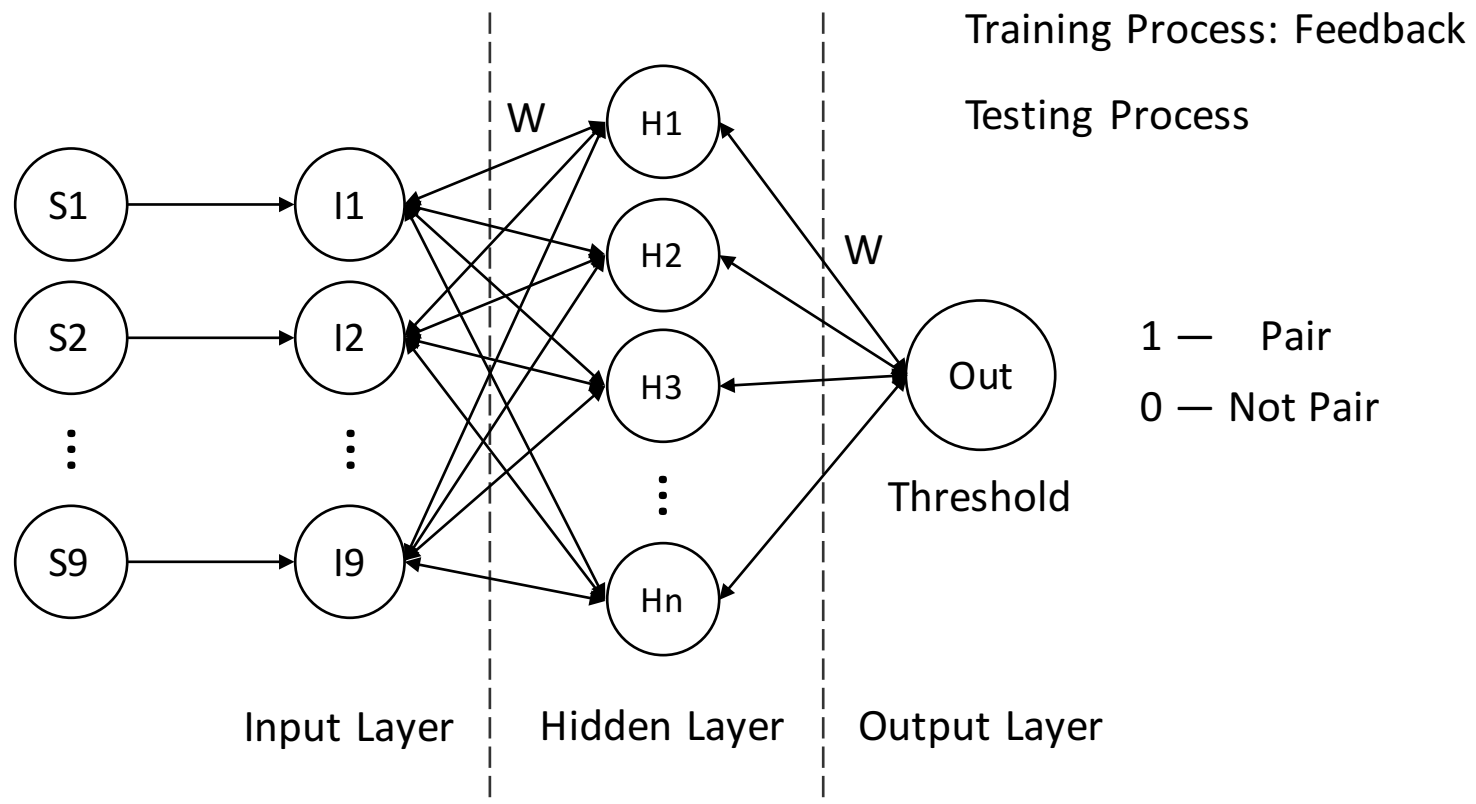


- Abstraction



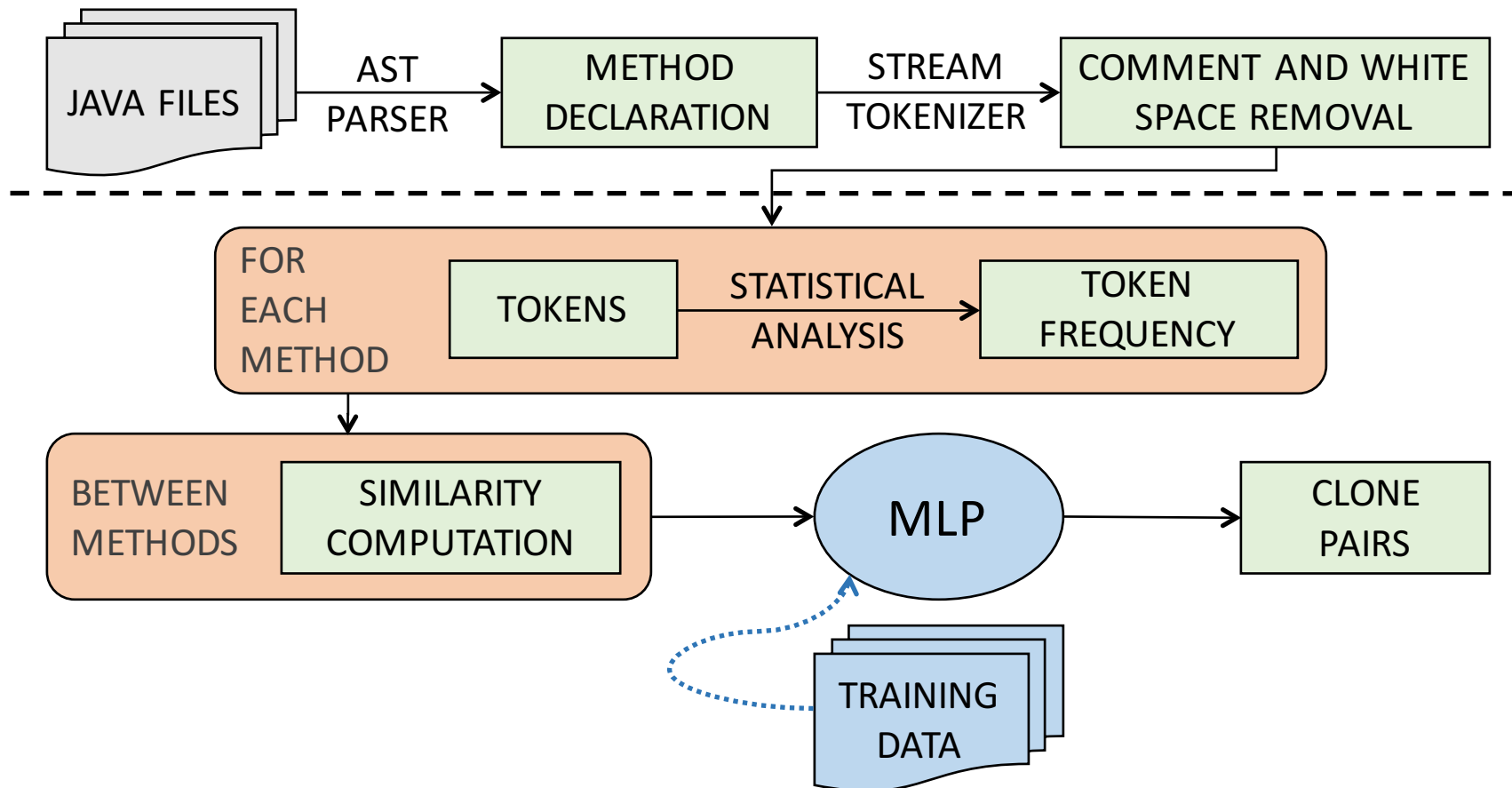
Technical Challenges

- Manual Weights
 - Multilayer perceptron (MLP)



Technical Challenges

- Manual Weights



Technical Challenges

- Challenges in mid-term presentation
 - ASTParser
 - Variables Comparison
 - Similarity Calculation
 - Manual Weights
 - Data Collection
 - Results Comparison
 - UI Development

X

✓

✓




✓

X

X

X

Technical Challenges

- Data Collection 
 - See Experiment Results
- Results Comparison 
 - See Experiment Results
- UI Development 
 - From Swing to SWT 4.5 release
 - Native elements and native behavior

Technical Challenges

- Challenges in mid-term presentation
 - ASTParser
 - Variables Comparison
 - Similarity Calculation
 - Manual Weights
 - Data Collection
 - Results Comparison
 - UI Development

X

✓

✓

✓

✓

✓

✓

Experiment Results

- Source: <https://github.com/CSCC5704>
 - ASTParserTool.java
 - BiGramSimilarity.java
 - CCDTool.java
 - MethodList.java
 - MethodSimilarity.java
 - MethodTokenizerTool.java
 - MethodVector.java
 - MultiplePerceptronTool.java
 - Result.java
 - TokenList.java
 - TokenVector.java

Experiment Results

- Detection of Software Clones

- <http://www.bauhaus-stuttgart.de/clones/>
- A general repository and information center for Detection of Software Clones
- Accept files with labeled clones for clone detection tool evaluation
- An example of submission file:

#1:	foo.c	12	56	bar.c	68	90	1
#2:	wom.c	34	50	bat.c	90	124	2
#3:	wom.c	69	100	bar.c	45	70	1
#4:	wom.c	69	100	foo.c	59	80	1

- Does not meet our requirement so we use manual selection to find ground truth

Experiment Results

- Training Data: SWT (# of methods 12~125)

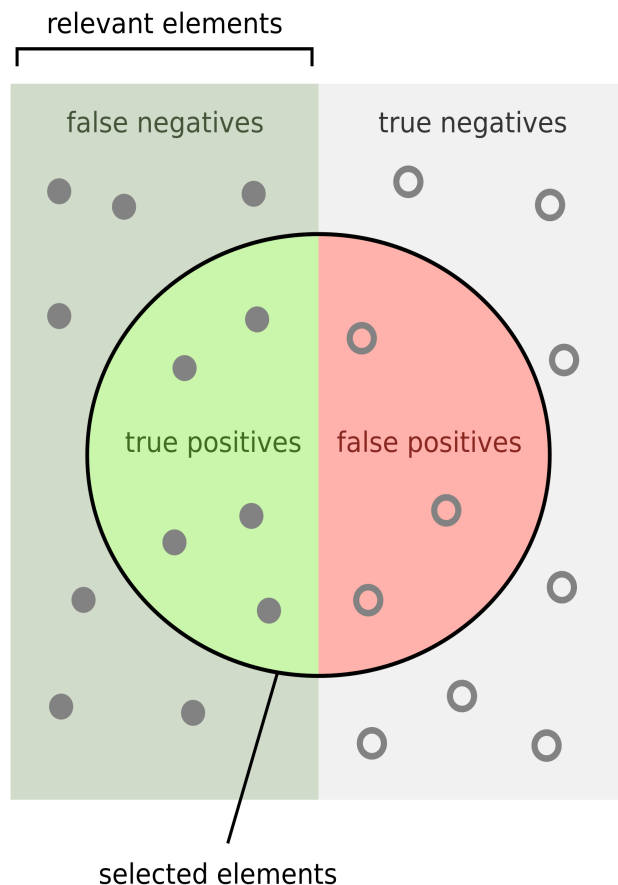
Combo.txt	Label.txt	Text.txt
DragSource.txt	Printer.txt	WebKit.txt
DropTarget.txt	Program.txt	
FormData.txt	Shell.txt	

- Test Data: SWT(# of methods 38~276)

Accessible.java	Menu.java	ToolBar.java
Button.java	Spinner.java	ToolItem.java
Control.java	TabFolder.java	Tracker.java
CoolBar.java	Table.java	Tree.java
Display.java	TableItem.java	

Experiment Results

- Evaluation: Precision and Recall



How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

Experiment Results

- Experiment 1

Test Files	TP + FN (Actual Clones)	TP + FP (Clones Detected)	TP
Button.java	0	4	0
CoolBar.java	11	12	10
Menu.java	1	0	0
Spinner.java	2	2	2
TabFolder.java	3	2	2
TableItem.java	20	20	18
ToolBar.java	4	4	4
ToolItem.java	3	3	3
Tracker.java	1	1	1
Tree.java	11	13	9

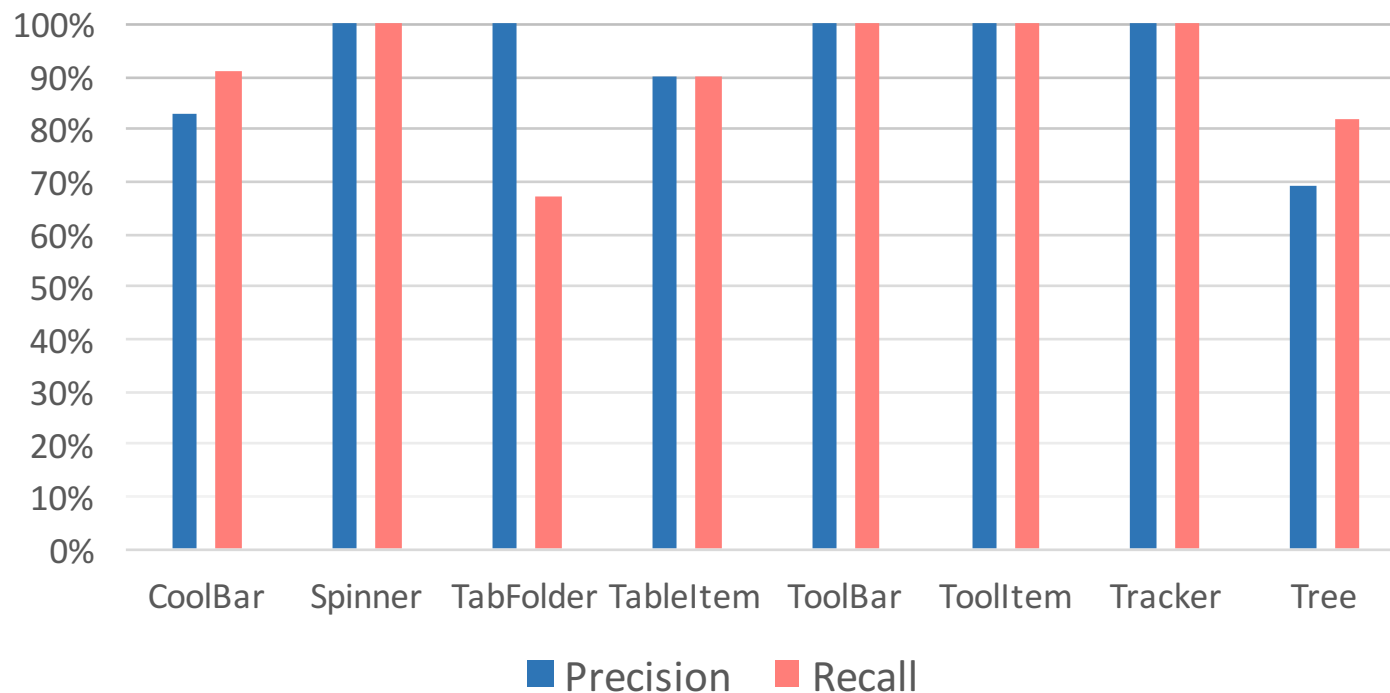
Experiment Results

- Experiment 1

Test Files	TP + FN (Actual Clones)	TP + FP (Clones Detected)	TP
Button.java	0	4	0
CoolBar.java	11	12	10
Menu.java	1	0	0
Spinner.java	2	2	2
TabFolder.java	3	2	2
TableItem.java	20	20	18
ToolBar.java	4	4	4
ToolItem.java	3	3	3
Tracker.java	1	1	1
Tree.java	11	13	9

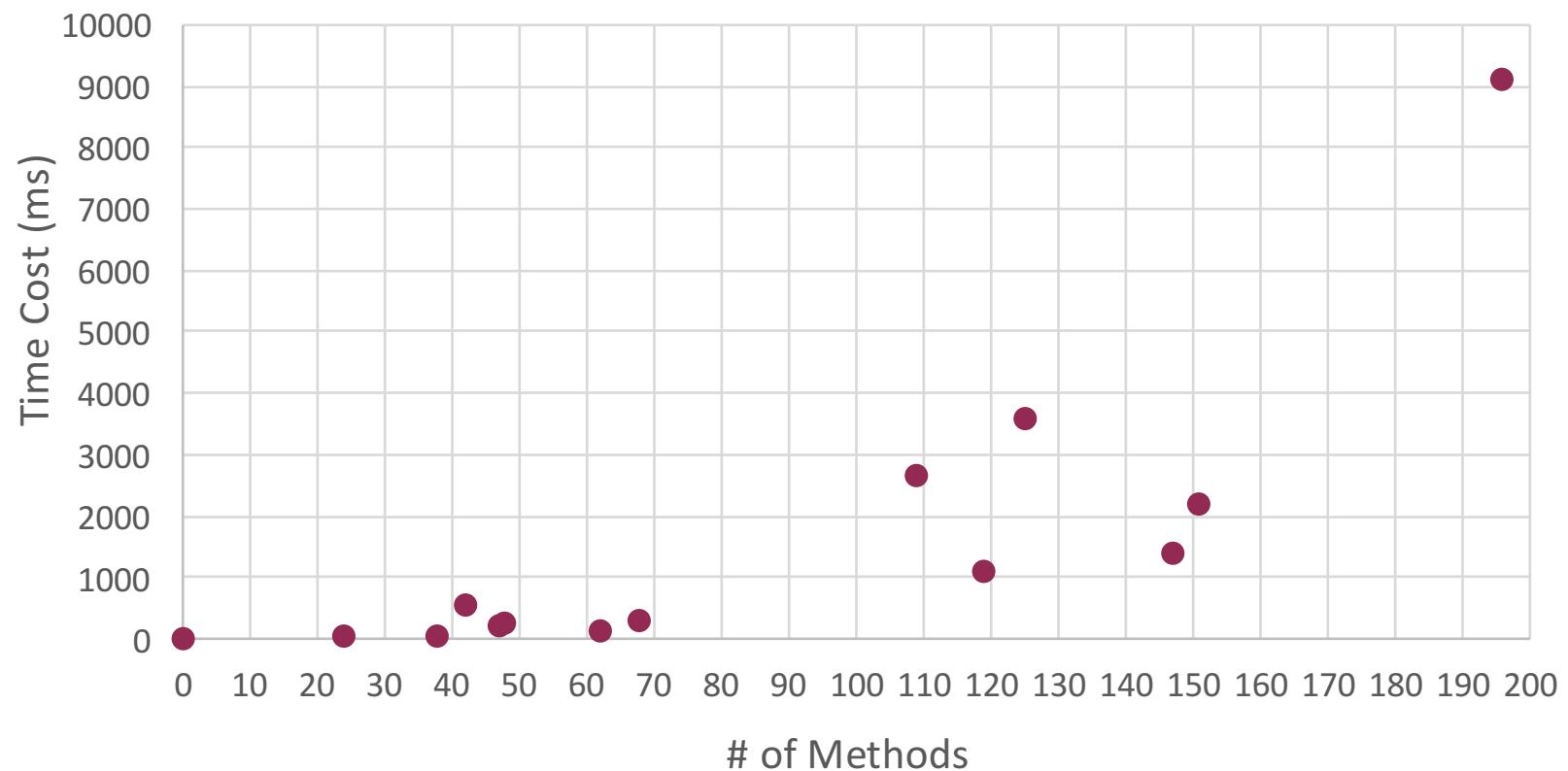
Experiment Results

- Experiment 1
 - Avg. Precision = 92.75%
 - Avg. Recall = 91.25%



Experiment Results

- Experiment 2



Things We Learn

- Lectures
 - ASTParser Tool
 - Bigram Algorithm
 - Precision and Recall
- Papers
 - Similarity Algorithms
 - Other Detection Methods
 - CMCD, Boreas, AnDarwin, RTF
- Others
 - First time to use SWT
 - Collaboration and discussion

Thank you !