Smelling Source Code Using Deep Learning

Tushar Sharma

http://www.tusharma.in


Refactoring
Designite
Open source
SE research
Java
Code smells
Public speaking
Books
Training
Architecture
IaC
Code quality
Python
Software design
Tools
C#
Change impact analysis


Smells

What is a smell?

… certain structures in the code that suggest (sometimes they scream for) the possibility of refactoring.

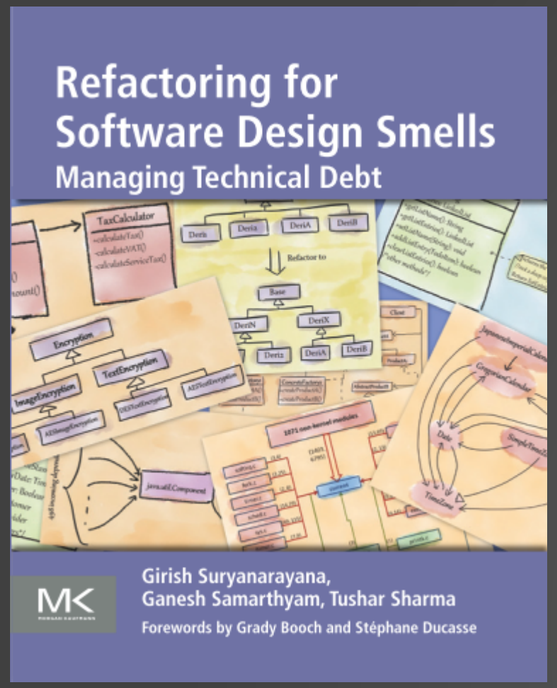
- Kent Beck

20 Definitions of smells: http://www.tusharma.in/smells/smellDefs.html

Smells' catalog: http://www.tusharma.in/smells/

Implementation smells

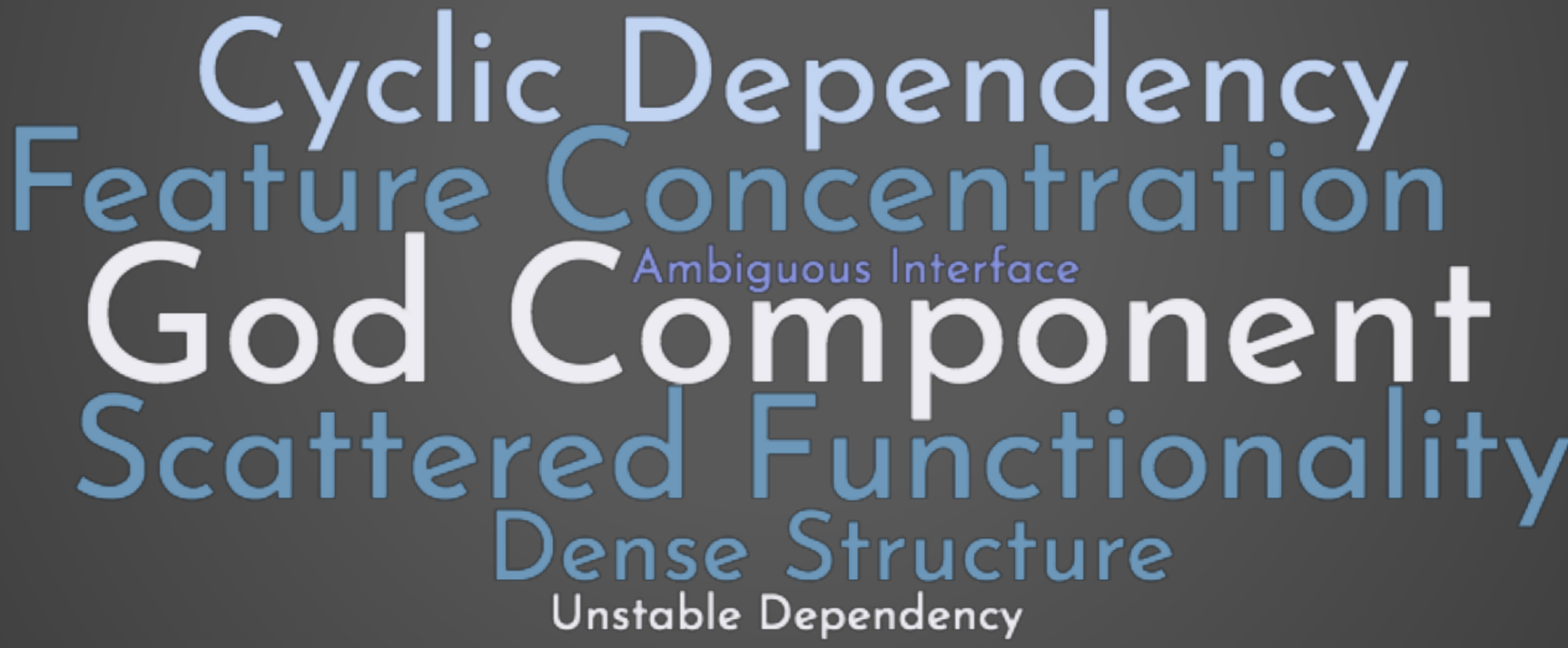
Duplicate Code
Long Parameter List
Complex Conditional
Complex Method
Magic Number
Empty Catch Block
Long Method
Missing Default
Long Identifier
Long Statement



Design Smells

Wide Hierarchy
Multipath Hierarchy
Rebellious Hierarchy
Speculative Hierarchy
Duplicate Abstraction
Unutilized Abstraction
Unexploited Encapsulation
Cyclically-dependent Modularization
Insufficient Modularization
Hub-like Modularization
Unnecessary Abstraction
Deficient Encapsulation
Incomplete Abstraction
Broken Modularization
Unfactored Hierarchy
Deep Hierarchy
Cyclic Hierarchy
Missing Abstraction
Multifaceted Abstraction
Imperative Abstraction
Unnecessary Hierarchy
Missing Encapsulation
Missing Hierarchy
Broken Hierarchy
Leaky Encapsulation

Architecture Smells



How smells get detected?

History-based

Heuristics-based

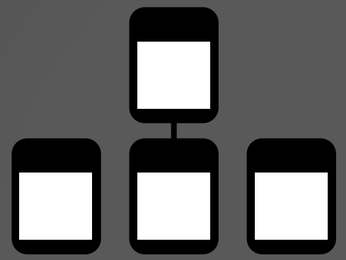
Metrics-based

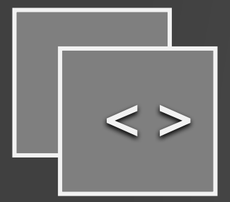
Machine learning-based

Optimization-based

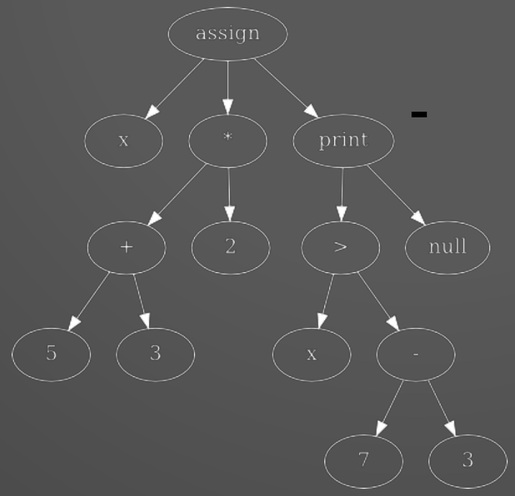
Metrics-based smell detection

Source model

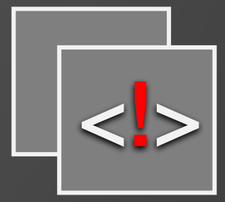




Code
(or source artifact)

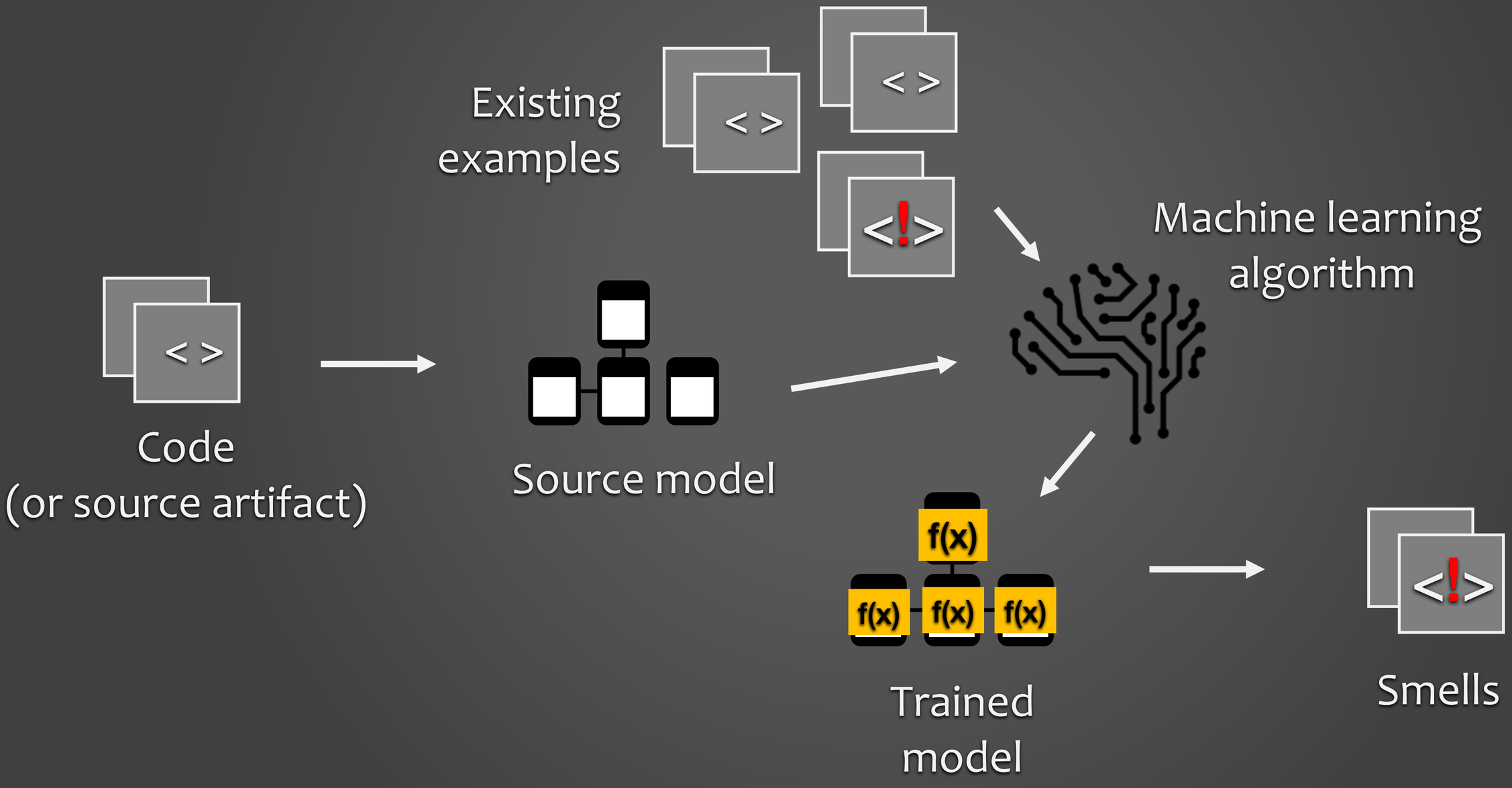


Metrics



Smells

Machine learning-based smell detection



Machine learning-based smell detection

Existing academic work:

- Support vector machines
- Bayesian belief network
- Logistic regression
- CNN

Take metrics as the features/input

Validation on balanced samples

Research questions

RQ1: Would it be possible to use deep learning methods to detect code smells?

RQ2: Is transfer-learning feasible in the context of detecting smells?

Transfer-learning refers to the technique where a learning algorithm exploits the commonalities between different learning tasks to enable knowledge transfer across the tasks

Overview

Research questions

GitHub

C# C# C#

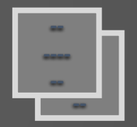
Java Java Java

Designite

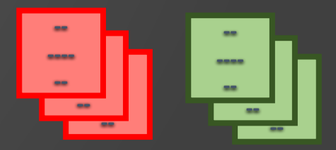
CodeSplit

Detected smells

Code fragments



Learning data generator



Positive and negative samples

Tokenizer

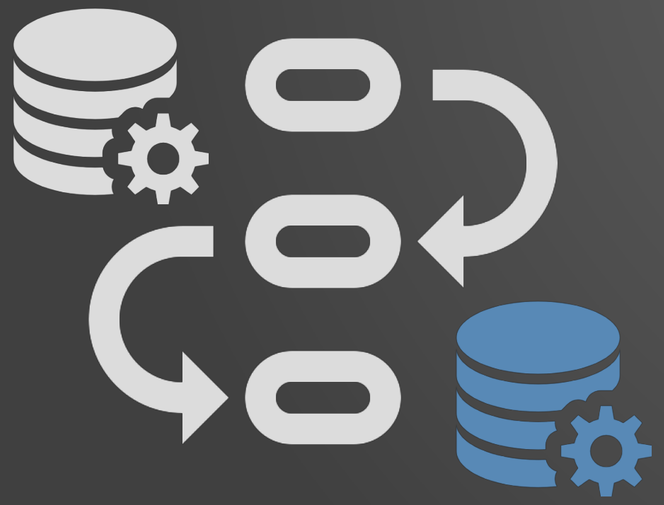
01 0110 0001 0110 1

Tokenized samples

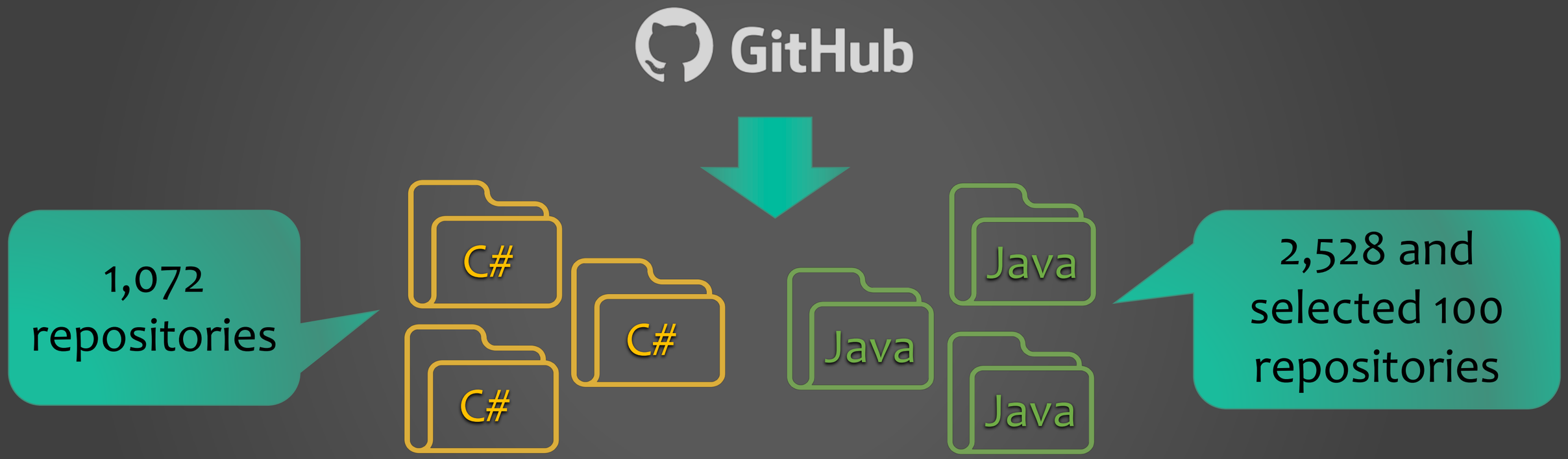
Preprocess

Deep learning models

Data Curation

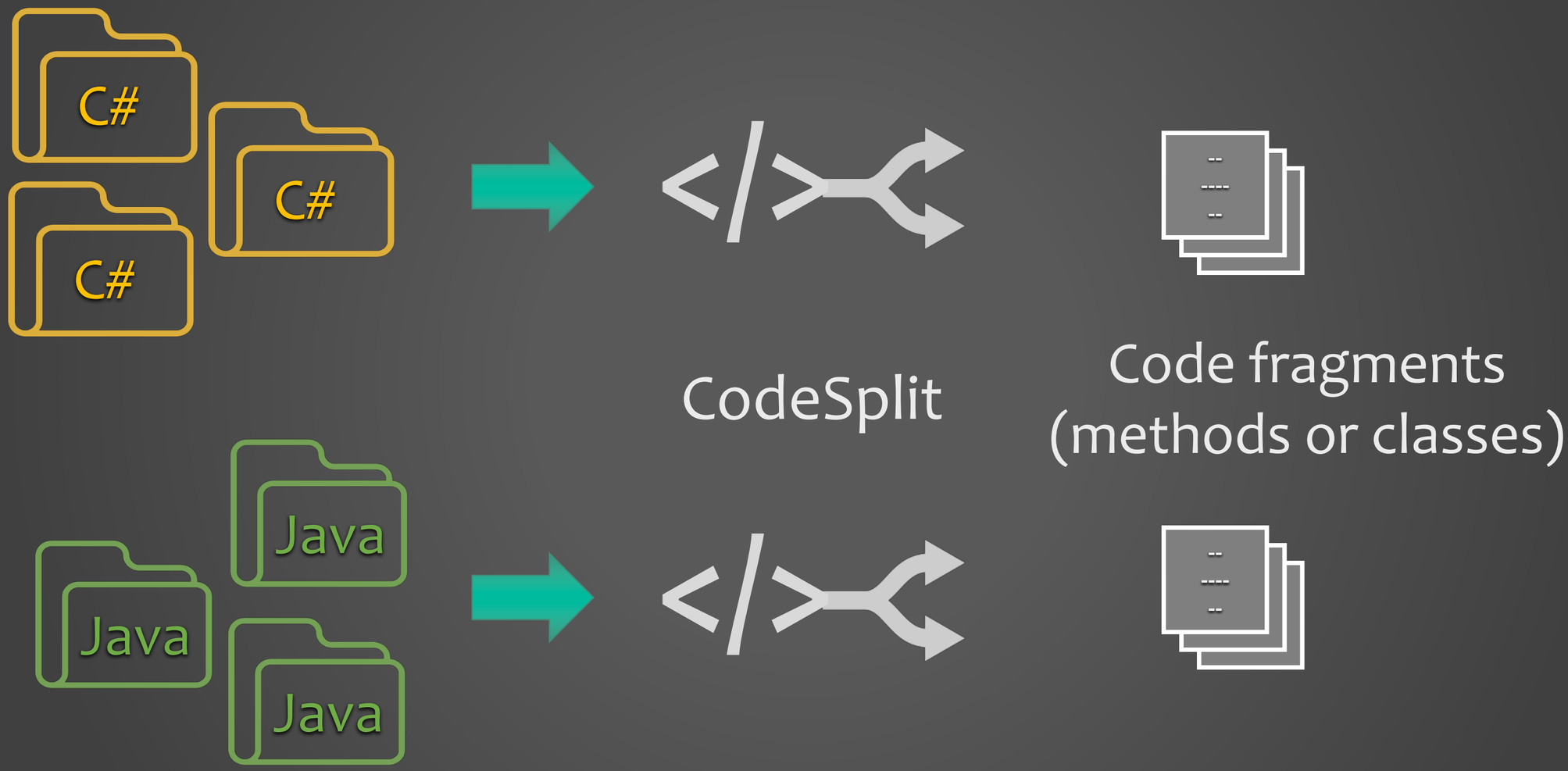


Repositories download



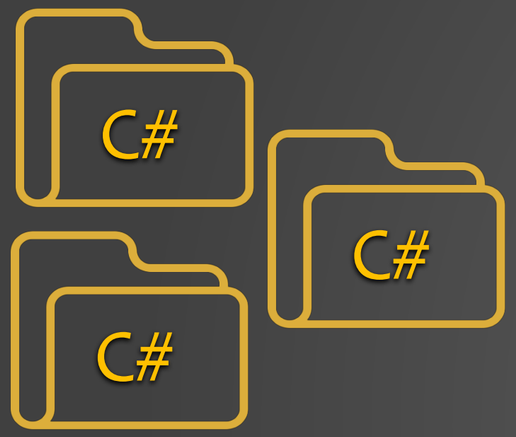
Architecture	Community	CI	Documentation	History	License	Issues	Unit test	Stars
✓	✗	✓	✓	✓	✗	✓	✓	

Splitting code fragments

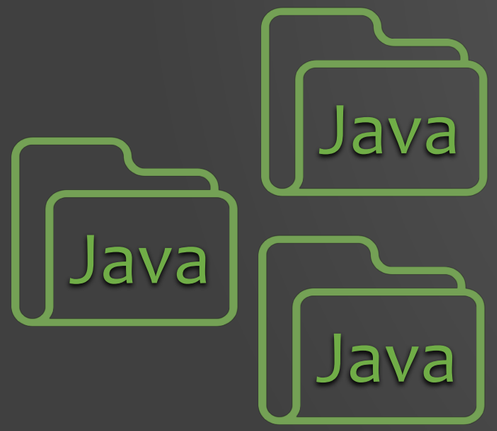


https://github.com/tushartushar/CodeSplitJava

Smell detection



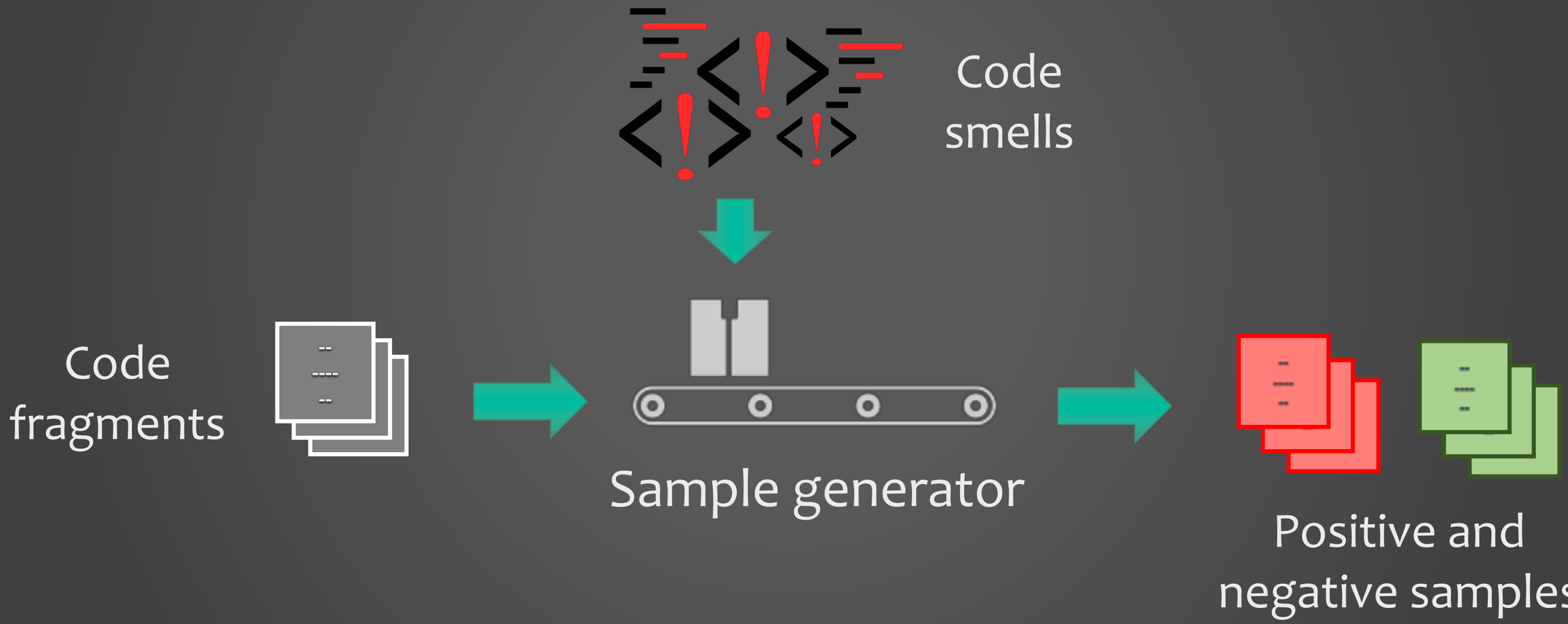
Detected code smells



http://www.designite-tools.com/

https://github.com/tushartushar/DesigniteJava

Generating training and evaluation samples



Tokenizing learning samples

Code fragments

Tokenizer

Tokenized samples

https://github.com/dspinellis/tokenizer

Tokenizing learning samples

```
public void InternalCallback(object state)
{
   Callback(State);
   try
   {
      timer.Change(Period, TimeSpan.Zero);
   }
   catch (ObjectDisposedException)
    {}
}
```

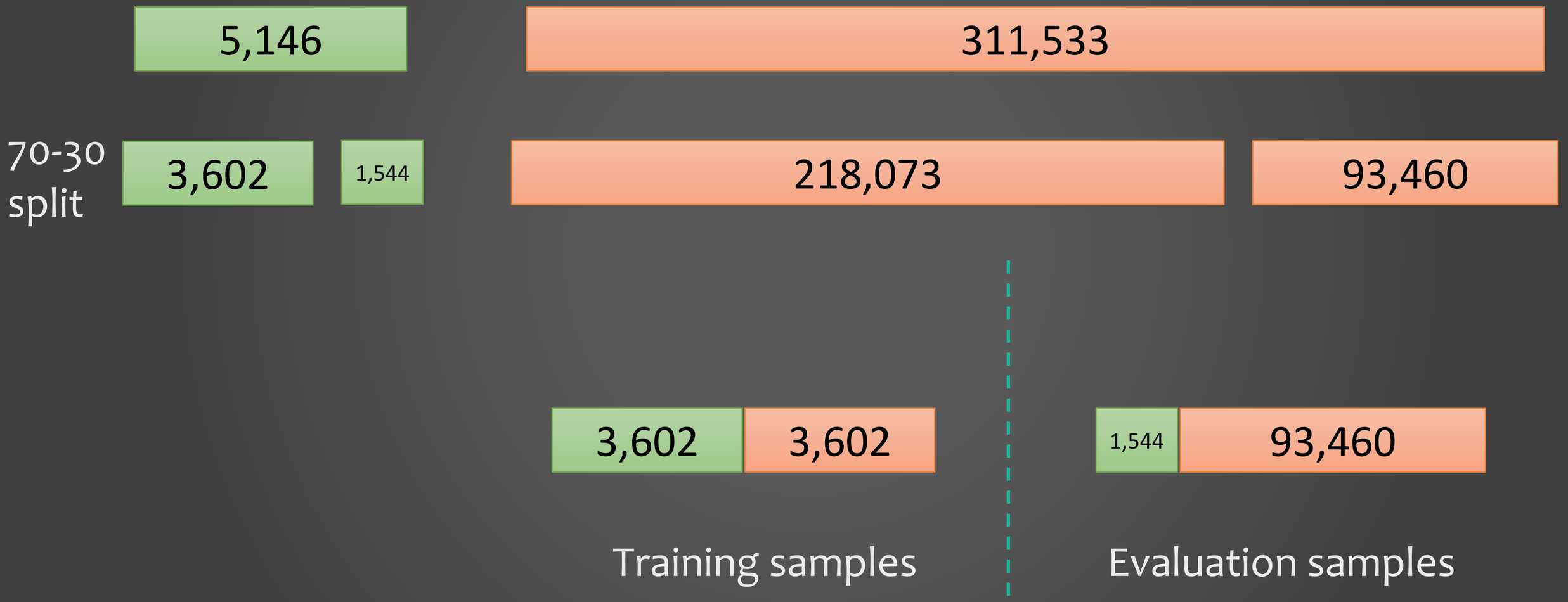
1-D

123 2002 40 2003 41 59 474 123 2004 46 200

2-D

123 2002 40 2003 41 59
474 123 2004 46 2005 40 2006 44 2007 46 2
125 329 40 2009 41 123 125 125

Data preparation



Selection of smells

- Complex method
- Magic number
- Empty catch block
- Multifaceted abstraction

the method has high cyclomatic complexity

an unexplained numeric literal is used in an expression

a class has more than one responsibility assigned to it

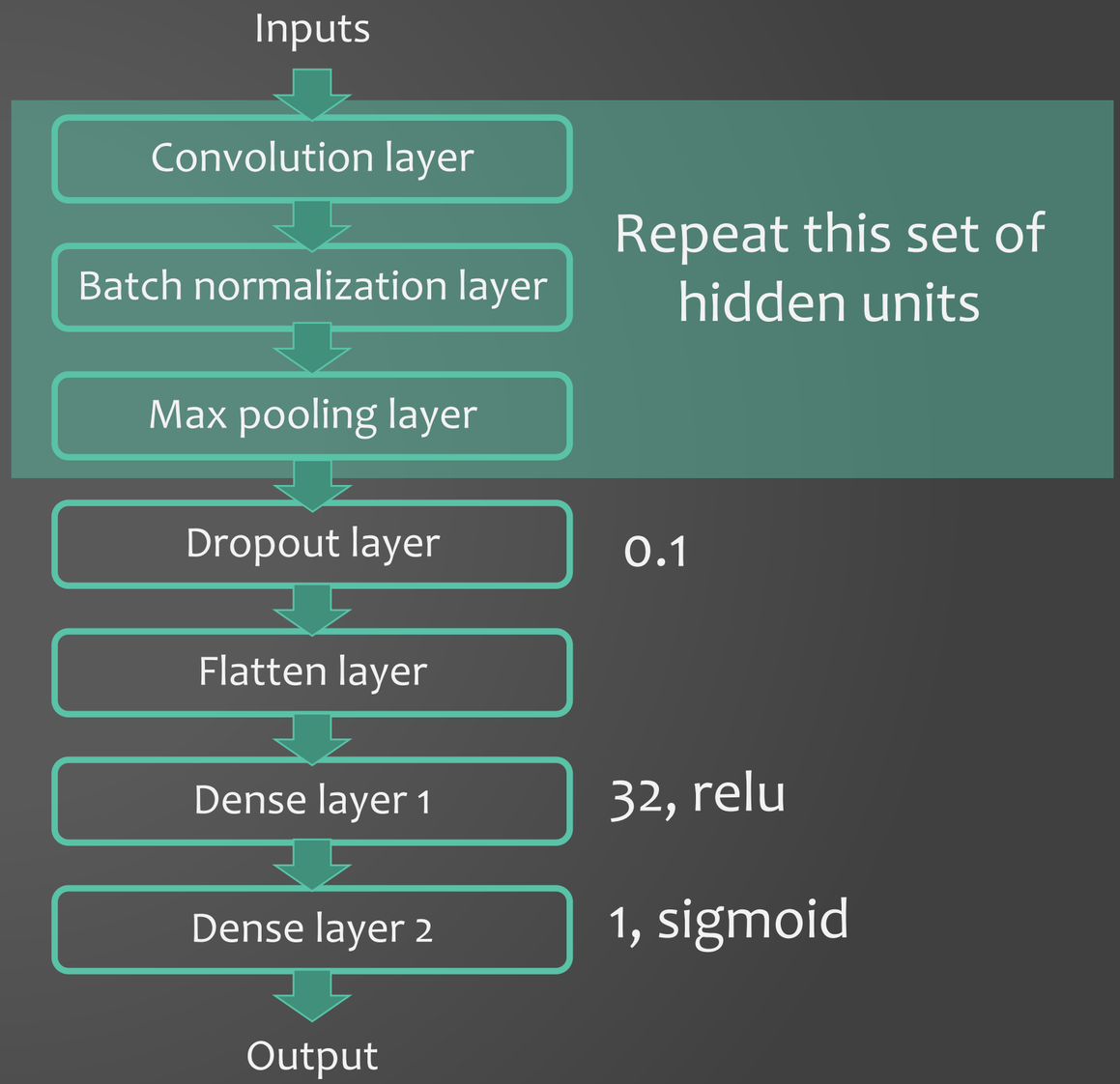
a catch block of an exception is empty

Architecture - CNN

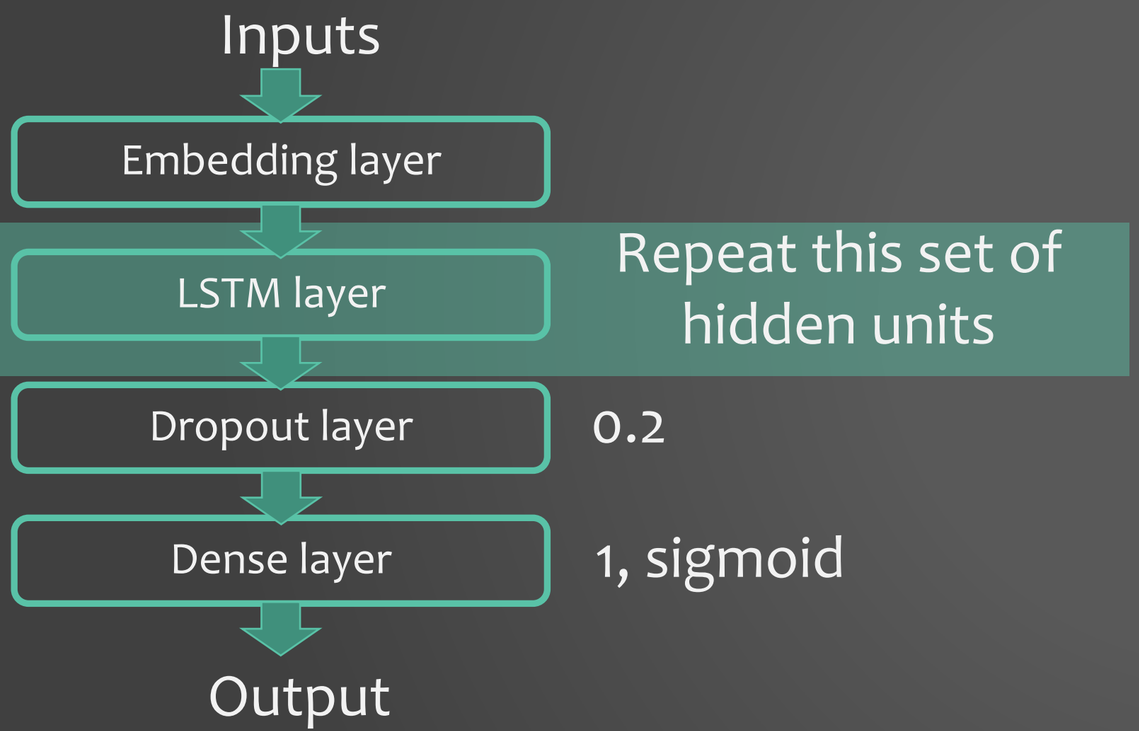
- Filters = {8, 16, 32, 64}
- Kernel size = {5, 7, 11}
- Pooling window = {2, 3, 4, 5}

- Dynamic Batch size = {32, 64, 128, 256}

- Callbacks
  - Early stopping (patience = 5)
  - Model check point



Architecture - RNN



- Dimensionality of embedding layer = {16, 32}
- LSTM units = {32, 64, 128}
- Dynamic Batch size = {32, 64, 128, 256}
- Callbacks
  - Early stopping (patience = 2)
  - Model check point

Running experiments

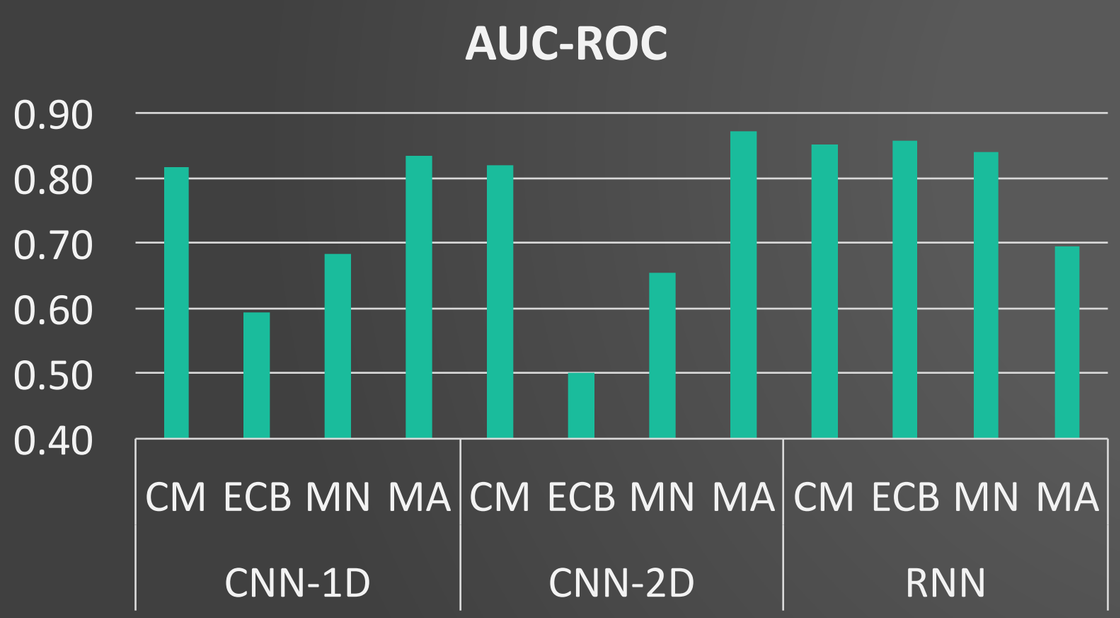
- Phase 1 – Grid search for optimal hyper-parameters
  - Validation set – 20%
  - Number of configurations
    - CNN = 144
    - RNN = 18
- Phase 2 – experiment with the optimal hyper-parameters

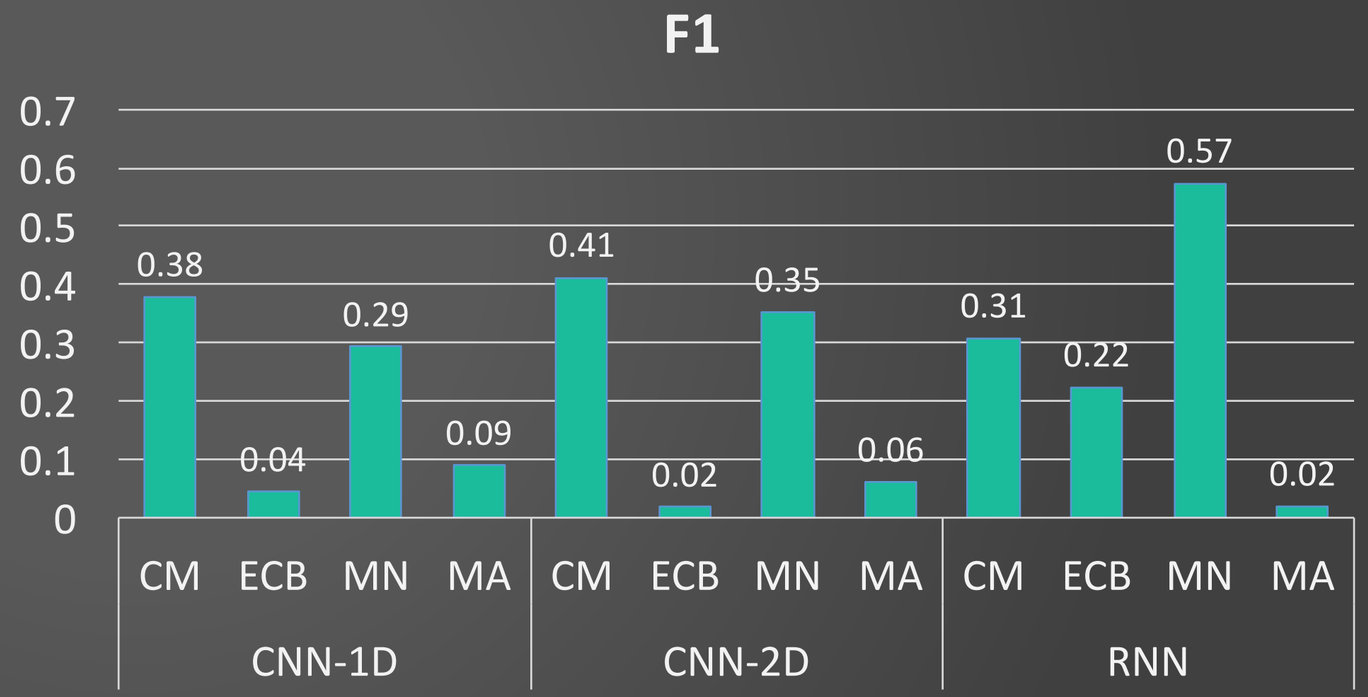
GRNET Super computing facility

Each experiment using 1 GPU with 64 GB memory

Results

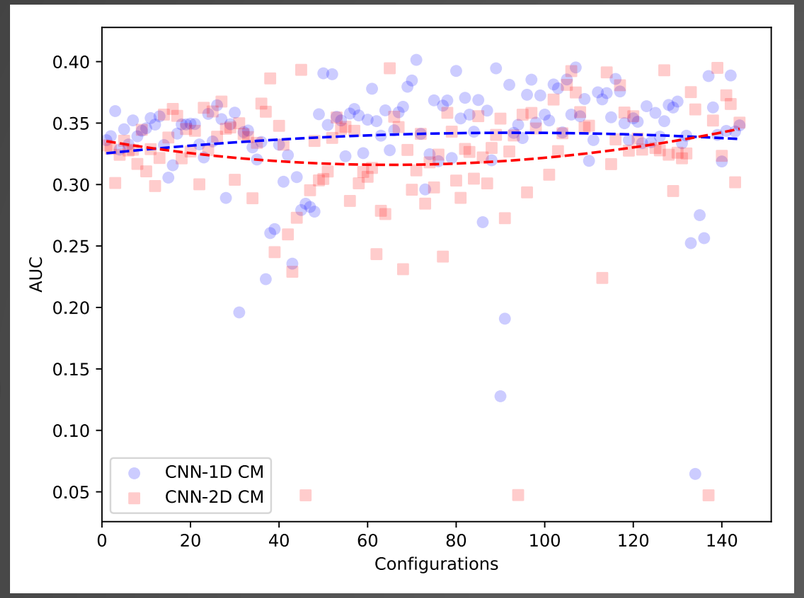
RQ1. Would it be possible to use deep learning methods to detect code smells?

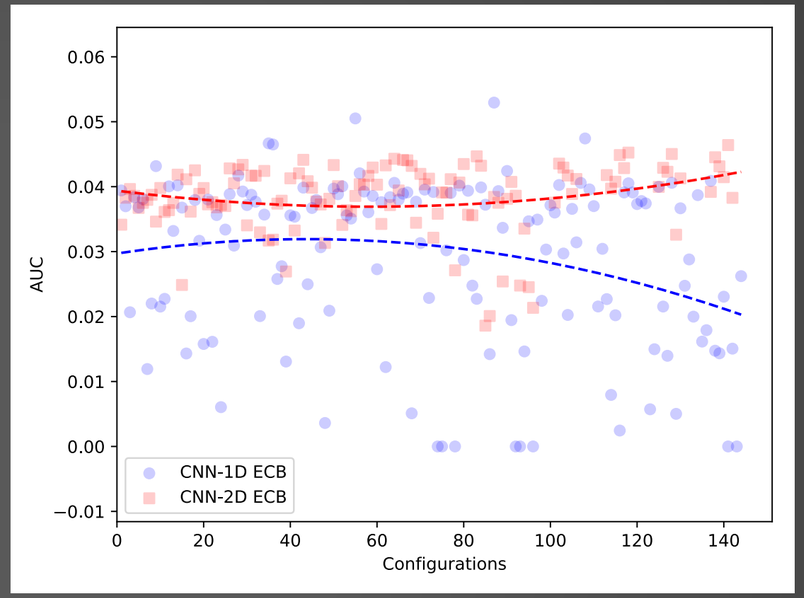




CNN-1D vs CNN-2D

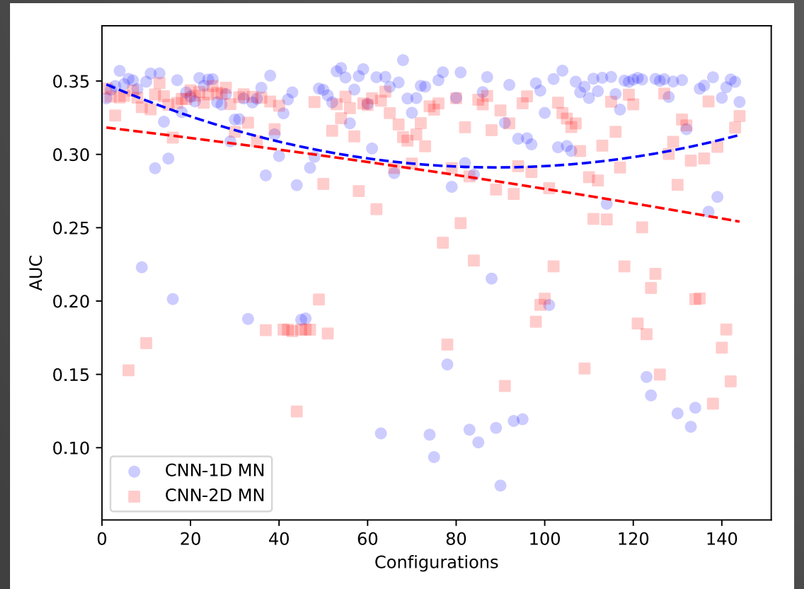
CNN-1D (max) - 0.40
CNN-2D (max) - 0.39

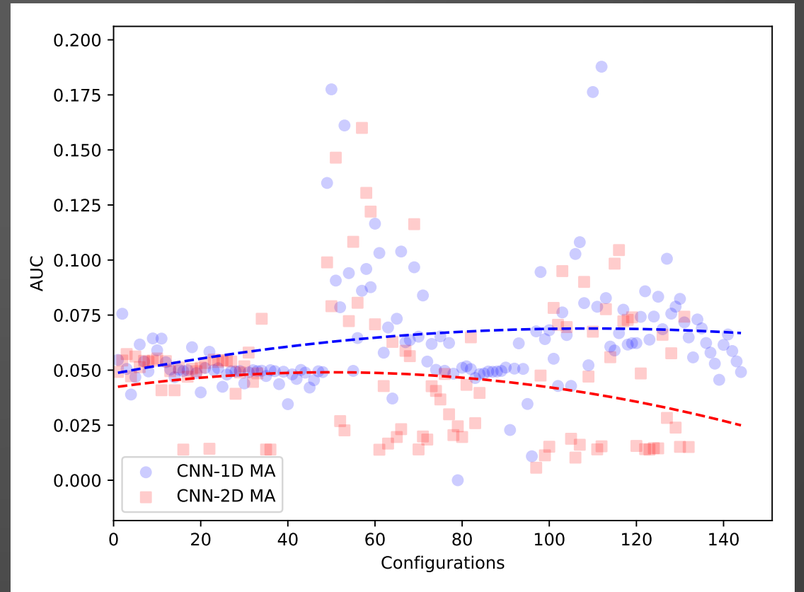




CNN-1D (max) - 0.05
CNN-2D (max) - 0.04

CNN-1D (max) - 0.36
CNN-2D (max) - 0.35





CNN-1D (max) - 0.18
CNN-2D (max) - 0.16

CNN vs RNN

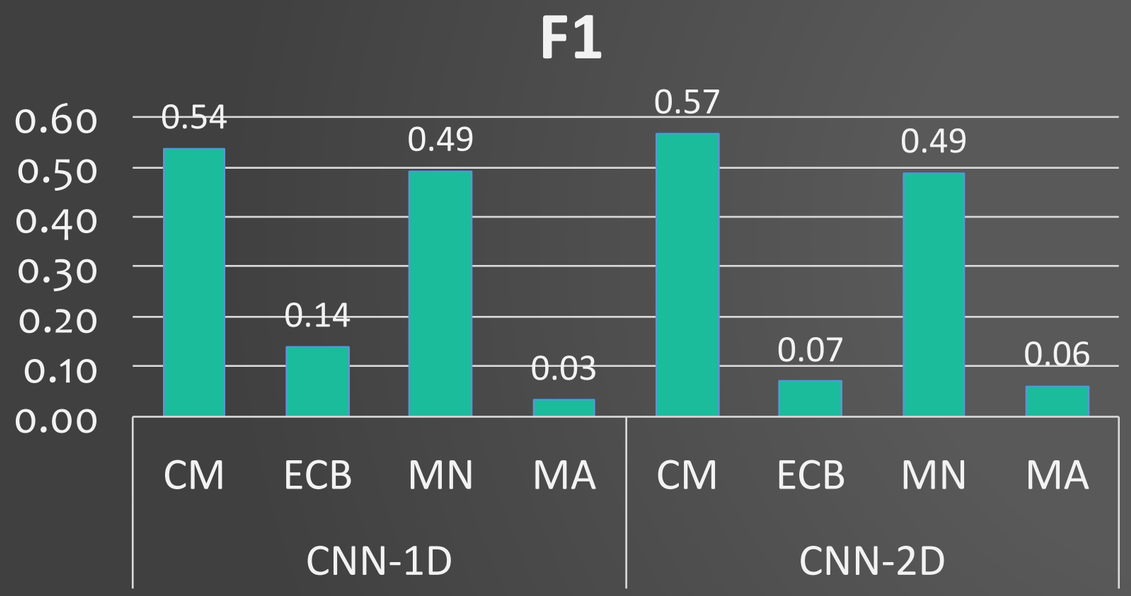
	RNN and CNN-1D	RNN and CNN-2D
CM	-22.94	-33.81
ECB	80.23	91.94
MN	48.96	38.58
MA	-349.12	-205.26

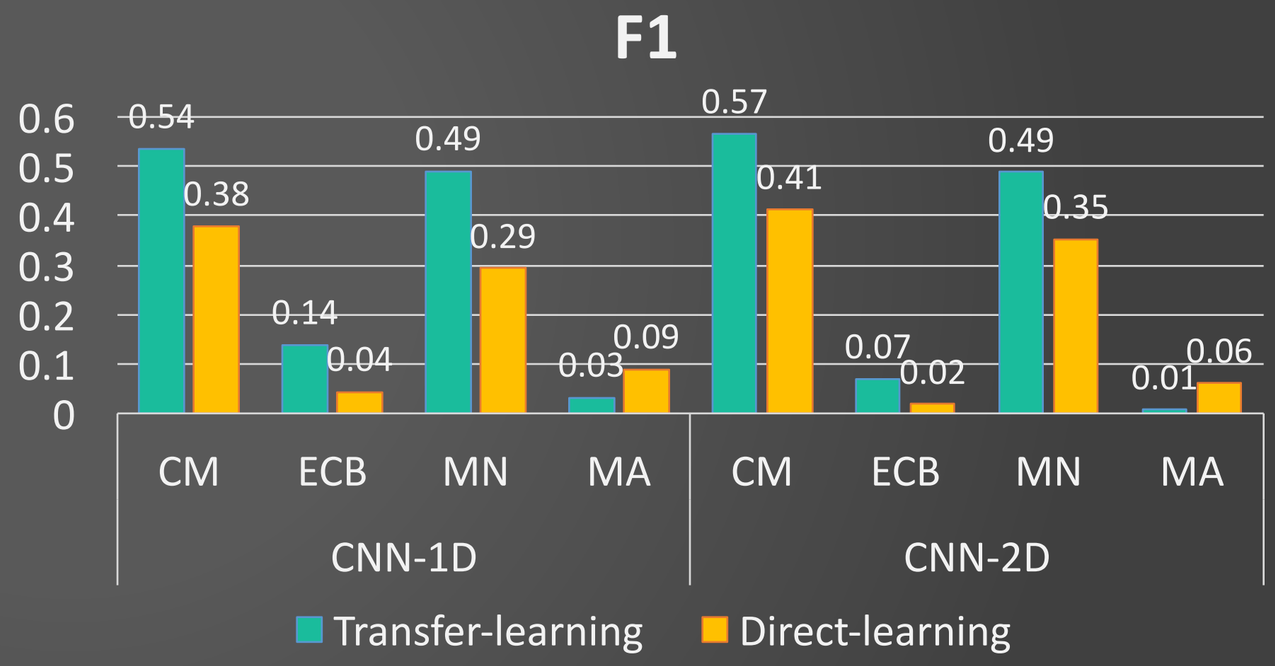
Difference in percentage; comparing max F1

Are more deep layers always good?

	Layers	CM	ECB	MN	MA
CNN-1D	1	0.36	0.05	0.36	0.08
	2	0.40	0.05	0.36	0.18
	3	0.40	0.05	0.36	0.19
CNN-2D	1	0.39	0.04	0.35	0.07
	2	0.39	0.04	0.34	0.16
	3	0.39	0.05	0.34	0.10
RNN	1	0.34	0.21	0.48	0.28
	2	0.36	0.24	0.48	0.22
	3	0.37	0.23	0.48	0.20

RQ2: Is transfer-learning feasible in the context of detecting smells?





Conclusions

It is feasible to make the deep learning model learn to detect smells

Transfer-learning is feasible.

Improvements – many possibilities
- Performance
- Add more smells – different kinds

Relevant links

Source code and data

https://github.com/tushartushar/DeepLearningSmells

Smell detection tool

Java - https://github.com/tushartushar/DesigniteJava

C# - http://www.designite-tools.com

CodeSplit

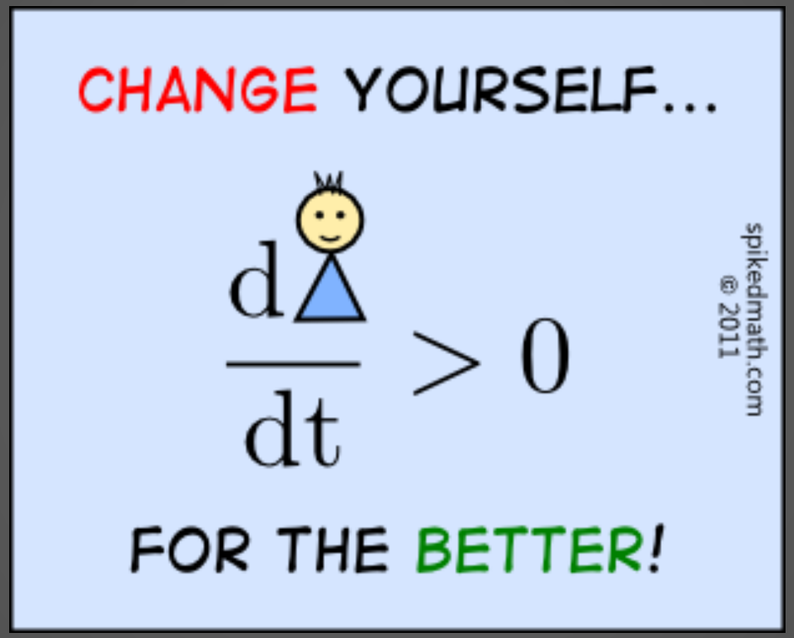
Java - https://github.com/tushartushar/CodeSplitJava

C# - https://github.com/tushartushar/DeepLearningSmells/tree/master/CodeSplit

Tokenizer

https://github.com/dspinellis/tokenizer

Thank you!!



Courtesy: spikedmath.com