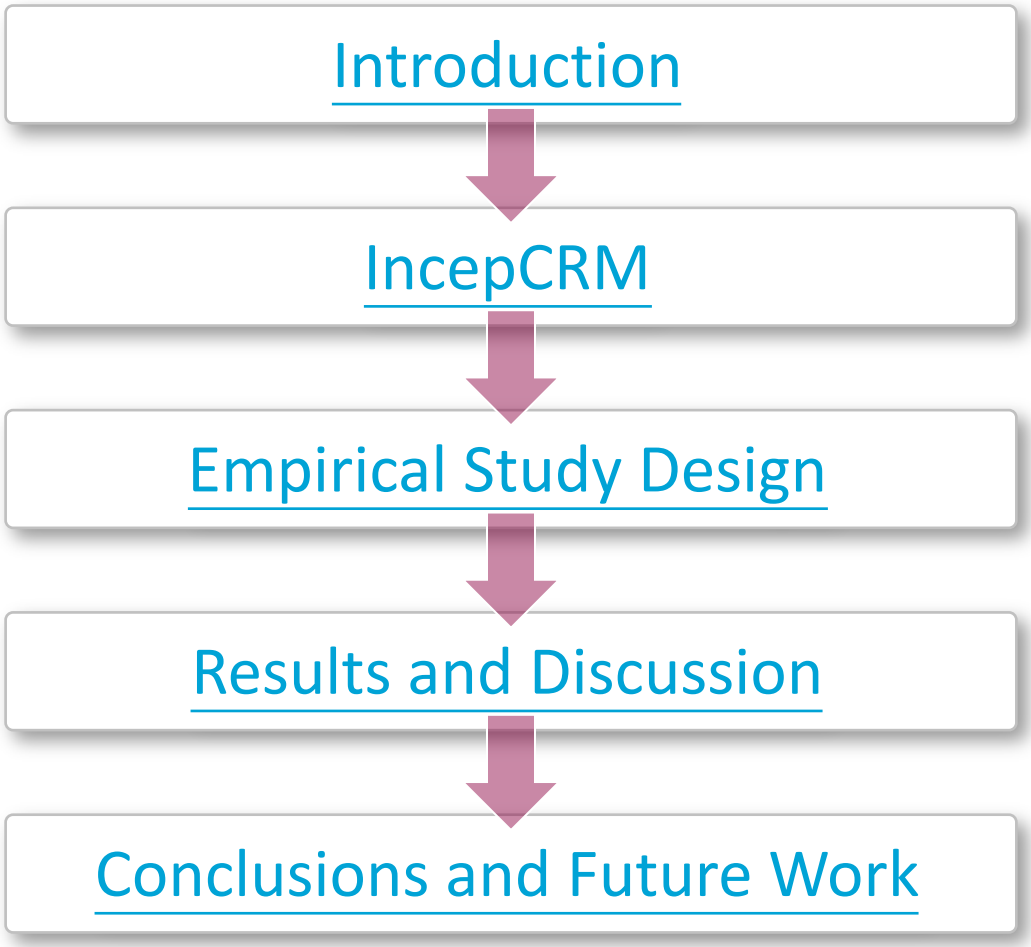
An Inception Architecture-Based Model for Improving Code Readability Classification

Qing Mi, Jacky Keung, Yan Xiao, Solomon Mensah, Xiupei Mei
Department of Computer Science
City University of Hong Kong
Kowloon, Hong Kong

CityU
香港城市大學
City University of Hong Kong

Department of
Computer Science

Presentation Outline



Background

- Code Readability can be defined as a human judgment of how easily a piece of code can be read and understood.
- As shown in Figure 1, it is desirable to have readable code that is less erroneous, more reusable, and easier to maintain.

```java
/**
 * Add one zero if necessary
 * @param number
 * @return
 */
private CharSequence addZero(int number) {
  StringBuilder builder = new StringBuilder();

  if (number < 10) {
    builder.append('0');
  }

  builder.append(Integer.toString(number));
```

Readable Code (Mean Readability Score - 4.02)

```java
  /**
   * Creates a new <code>DisbandUnitAction</code>.
   *
   * @param freeColClient The main controller object for the client.
   */
  DisbandUnitAction(FreeColClient freeColClient) {
      super(freeColClient, "unit.state.8", null, KeyStroke...
      putValue(BUTTON_IMAGE, freeColClient.getImageLibrary()...
      ImageLibrary.UNIT_BUTTON_DISBAND,
              0));
      putValue(BUTTON_ROLLOVER_IMAGE, freeColClient...
      getUnitButtonImageIcon(
              ImageLibrary.UNIT_BUTTON_DISBAND, 1));
```

Unreadable Code (Mean Readability Score - 2.01)

Figure 1: Readable Code vs. Unreadable Code

Introduction

❖ Traditional Models

Collect labeled data by conducting a large-scale survey, inviting multiple human annotators to rate code snippets by readability.

Handcraft features from different aspects that intuitively seem to have some effect on code readability.

Train a machine learning classifier using data collected from the first phase.

VS

❖ IncepCRM

Eliminate the need for manual feature engineering.

Apply the information of human annotators as the auxiliary input for model training.

Propose a modified Inception module that can automatically learn multi-scale features from the source code.

Convolutional Neural Network

- Feature Learning Network
  - Alternating convolutional and pooling layers
- Classification Network
  - One or more fully-connected layers

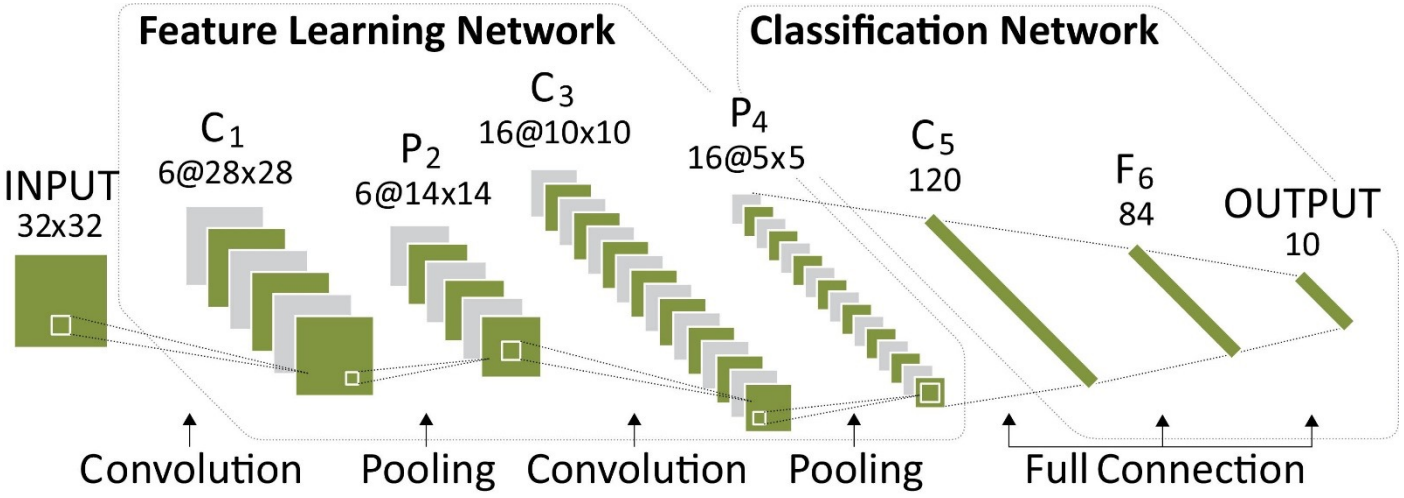


Figure 2: Architecture of LeNet-5 (A Classical Convolutional Neural Network)

Proposed Approach

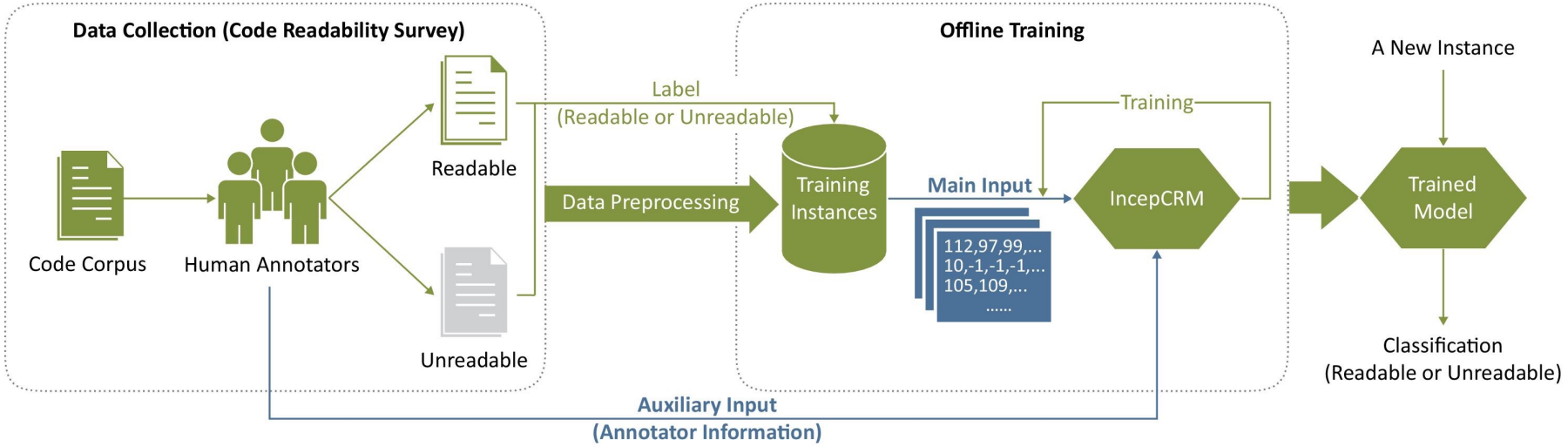
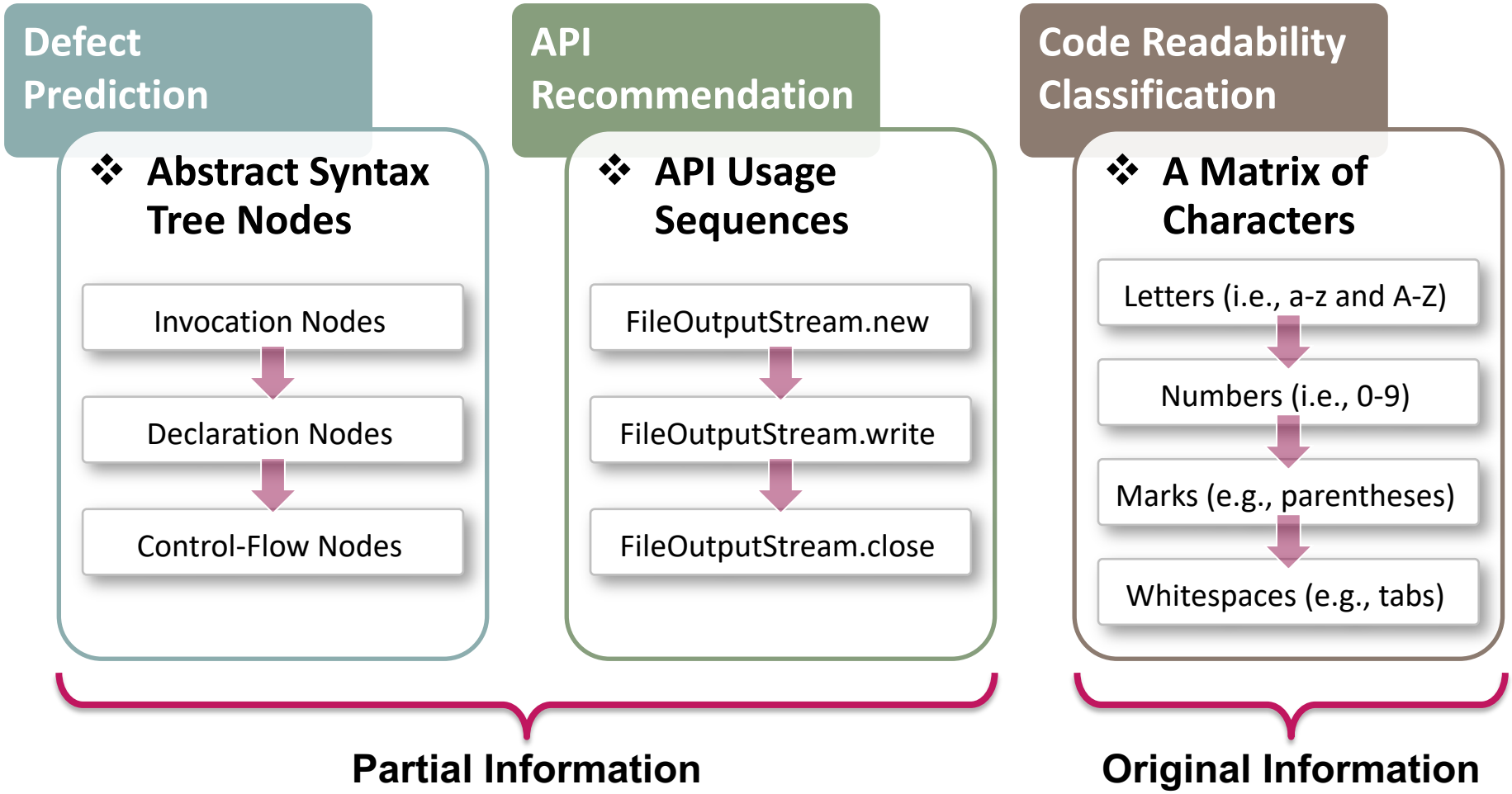


Figure 3: Approach Overview

Data Preprocessing



Annotator Information

- The number of annotators who have rated a code snippet
  - Code snippets that have been rated by more annotators are more reliable than those with less.
- The standard deviation of readability scores on a code snippet
  - A lower standard deviation indicates greater consistency among annotators.

Model Architecture

- Inspired by GoogLeNet, we adapt the Inception architecture for code readability classification.

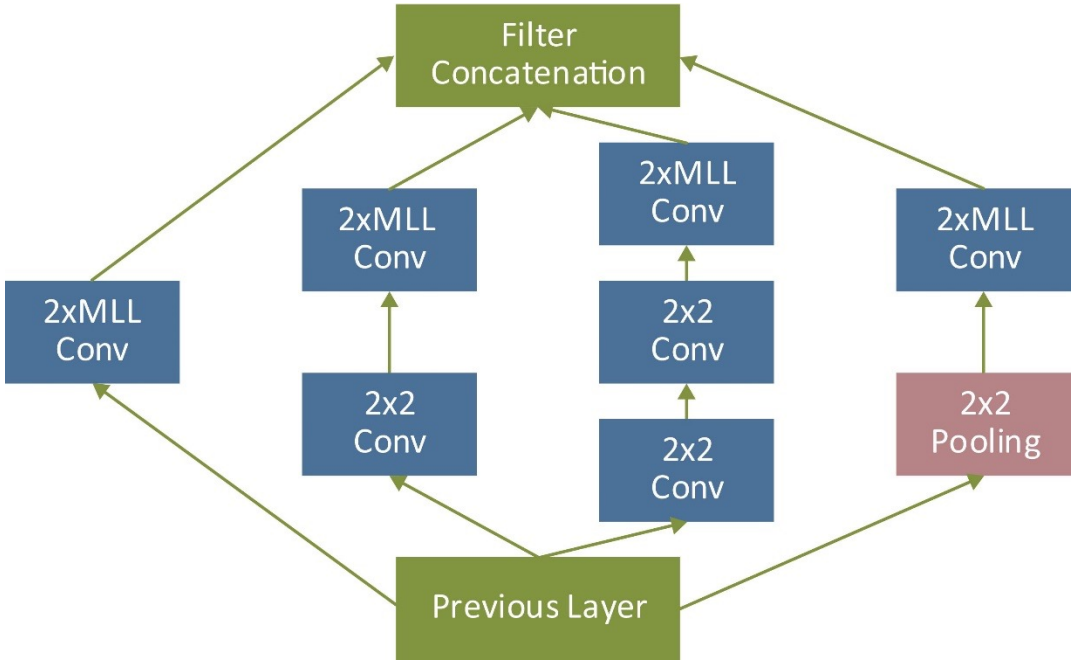


Figure 4: The Inception Module used in IncepCRM

- For computational savings, we apply the same receptive field in each module.
- The width of the filter size on top levels is set as the maximum line length (MLL).

Model Training

- We use the Keras library with Tensorflow backend to implement IncepCRM.
- The cross-entropy loss is adopted to calculate the error rate between actual and desired outputs.
  - N=32 is the number of data samples (the batch size)
  - M=2 is the number of classes (Readable and Unreadable)

$$J = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} p_j^{(i)} \log q_j^{(i)}$$

- We train the model in a supervised manner using the Adam optimizer with a learning rate of 0.001.

Empirical Study Design

- RQ1: Does annotator information contribute to the performance of IncepCRM?
- RQ2: To what extent does IncepCRM outperform the existing models in code readability classification?

Data Preparation

- A corpus of labeled code snippets as the ground-truth
- A Likert scale ranging from 1 (very unreadable) to 5 (very readable)
  - the Readable group VS the Unreadable group

Table 1: Statistical Summary of the Code Corpus

Dataset	Source	Total # of Code Snippets	Total # of Annotators	Lines of Code	
				Mean	SD
D_{Buse}	SourceForge	100	121	7.61	2.59
D_{Dorn}	SourceForge	360	5000+	29.74	16.36
$D_{Scalabrino}$	SourceForge	200	9	26.56	10.29

Measures for Evaluation

- Accuracy shows the ratio of all correctly classified instances.

$$Accuracy = \frac{tp + tn}{tp + fp + tn + fn}$$

- The higher the metric value, the better the model performance.

Analysis Method

- To answer RQ1, we examine whether there exists significant difference between the performance of IncepCRM with and without annotator information using robust test statistics.
  - The Brunner-Munzel test (significance level $\alpha=0.05$)
  - Cliff's delta

Analysis Method

- To answer RQ2, we compare IncepCRM with five state-of-the-art code readability models.
  - Buse et al.'s Model (M_{Buse})
  - Posnett et al.'s Model ($M_{Posnett}$)
  - Dorn et al.'s Model (M_{Dorn})
  - Scalabrino et al.'s Model ($M_{Scalabrino}$)
  - A Comprehensive Model (M_{All})
- To enable a fair comparison, we apply different machine learning techniques as the underlying classifier for each model.
  - Logistic Regression (LR), Bayesian Network (BN), Multilayer Perception (MLP), Sequential Minimal Optimization (SMO), and Random Forest (RF).

Results and Discussion

- RQ1 is to investigate whether the addition of annotator information can further improve the accuracy of code readability classification.
  - Negligible for |d-value| < 0.147, Small for |d-value| < 0.330, Medium for |d-value| < 0.474, and Large for otherwise.
- We observe that without annotator information, IncepCRM is able to correctly classify 80.19% of code snippets in our corpus. When annotator information is included, the accuracy increases to 82.76%.
  - The p-value for the Brunner-Munzel test is 0.016 (<0.05) and the d-value for Cliff's delta is 0.641 (>0.474), implying a significant difference with a large effect size.

Table 2: Accuracy Achieved by Code Readability Models

Model		D_{Buse}	D_{Dorn}	$D_{Scalabrino}$
M_{Buse}	LR	0.810	0.786	0.705
	BN	0.760	0.750	0.625
	MLP	0.760	0.742	0.665
	SMO	0.820	0.797	0.670
	RF	0.800	0.781	0.680
$M_{Posnett}$	LR	0.780	0.728	0.660
	BN	0.760	0.681	0.695
	MLP	0.770	0.703	0.655
	SMO	0.770	0.711	0.680
	RF	0.770	0.703	0.605
M_{Dorn}	LR	0.800	0.800	0.755
	BN	0.670	0.747	0.640
	MLP	0.720	0.725	0.685
	SMO	0.770	0.767	0.725
	RF	0.750	0.744	0.690
$M_{Scalabrino}$	LR	0.740	0.772	0.680
	BN	0.530	0.681	0.640
	MLP	0.770	0.742	0.665
	SMO	0.720	0.717	0.650
	RF	0.720	0.742	0.655
M_{All}	LR	0.790	0.839	0.795
	BN	0.720	0.761	0.710
	MLP	0.730	0.778	0.700
	SMO	0.810	0.806	0.770
	RF	0.770	0.789	0.690
IncepCRM		**0.825**	**0.842**	**0.822**

- IncepCRM yields higher accuracy against its competitors across all datasets.

Threats to Validity

- Internal Validity
  - Only a few labeled data are available in the literature, which may not be adequate to train IncepCRM.
  - If the data is prone to errors (e.g., wrong records), then it will be reflected in our results.
- External Validity
  - All projects investigated in this study are collected from SourceForge. It is unclear whether the same findings would be observed in other projects.
- Construct Validity
  - We mainly use accuracy to evaluate the effectiveness of IncepCRM. However, other performance measures such as F-Measure and AUC can also be considered.

Conclusions and Future Work

- We proposed IncepCRM, a novel approach to improve the accuracy of code readability classification based on the Inception architecture.
  - A modified Inception module suitable for code readability classification. (80.19%)
  - The use of annotator information as the auxiliary input for model training. (82.76%)

Conclusions and Future Work

- We conducted a series of experiments on three publicly available datasets.
  - (RQ1) Annotator information is beneficial for classification performance.
  - (RQ2) IncepCRM achieves state-of-the-art accuracy in comparison to previous code readability models.

Conclusions and Future Work

- Improve the performance of IncepCRM
  - Fine-tune hyperparameters (e.g., filter size and dropout rate)
  - Combine multiple deep learning models
  - Experiment with different configurations (e.g., loss functions)
- Find out what exact factors make a code more or less readable

Thank you for your attention!

For further enquires, please contact Qing Mi:
Qing.Mi@my.cityu.edu.hk

香港城市大學
City University of Hong Kong
Department of
Computer Science