

Why P not Q? Interpreting Code Model's Decision Making

Presenter:

Shamsa Abid

Research Scientist, RISE Lab, SCIS, SMU

iStock™
Credit: Phonlam

Core Research Areas

- Code Models Trustworthiness
- Code Models Interpretability
- Code Recommendation
- Code Clones
- Code Reuse
- Empirical Software Engineering
- LLM4Code
- ML4SE

What are we talking about today?

- AI Models Reliability and Trustworthiness
- Interpreting Code Models' Decision Making
- Semantic Code Clone Detection
- AI Systems Explainability and Interpretability
- Human vs Model Intuition for Code Clones
- Causal Interpretability
- Trustworthiness and Reliability Metrics
- Ranking Models
- Future Research Directions



Introduction



Need to understand model's
decision making

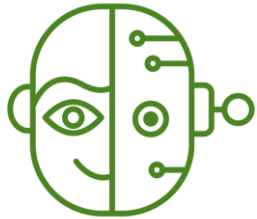


Why we need to move beyond
accuracy

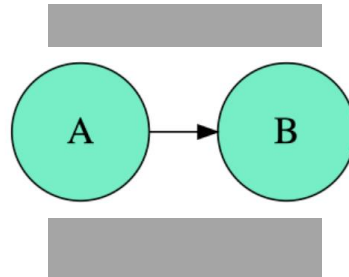


Why we need causal explanations
AND HOW TO GET THEM

Introduction



Our goal is to evaluate the performance of models in relation to human intuition.



Specifically, I will discuss how we apply counterfactual data mutations to get causal explanations



allows us to evaluate a model's reliability and trustworthiness.

Interpretability Overview

- *Real World Problems*
 - Criminal risk assessment tool, shows racial biases
 - European Union's "Right to Explanation"
- *Software Engineering Problems*
 - Explaining Predictions of Code Tasks
 - Semantic Code Clone Detection Task

Model	Reported F1-Score	Observed F1-Score
CodeBERT	94.0%	71.11% ↓
CodeGraph44CCDetector	96.6%	53.76% ↓
CodeT5	97.2%	65.9% ↓

Interpretability Overview

- Interpretability
 - the degree to which a human can understand the cause of a decision
 - Interpretability also defined as a part of *explainability*.
- Explainable models
 - summarize the reasons for neural network behaviors
 - gain the trust of the users
 - generate insights into the causes of their decisions.
 - *“You were denied a loan because your annual income was £30,000. If your income had been £45,000, you would have been offered a loan.”*

A Story of Misprediction



Semantic Code Clones

```
1 def is_sorted(stuff) :  
2   for i in range(1, len(stuff)) :  
3     if stuff [i - 1] > stuff [i] :  
4       return False  
5   return True
```

Listing 1: Clone208.py - A

```
1 def is_sorted(stuff) :  
2   for index, item in enumerate(stuff) :  
3     try :  
4       if item > stuff [index + 1] :  
5         return False  
6   except IndexError :  
7     return True
```

Listing 2: Clone208.py - B

Applications of code clones

SE Problems	Desired Solution	Goals
Hard to maintain duplicate code instances	Automatic change propagation or deduplication	Improved maintainability
Code has poor performance	Recommend functionally similar but optimized code variants	Code optimization
Re-inventing code	Abstraction of repeating functionality to a library	Code reuse Improved developer productivity
Repetitive code search	Provide related code on-the-go	Opportunistic code reuse Improved developer productivity

Concerns with Clone Detection Models



What parts of code in a clone pair is a model looking at to classify it as a semantic clone?



How far or close the model's code comprehension behavior is to human's code comprehension?



How can we identify causes of mispredictions?

Misprediction Analysis

- True Clone pair predicted as False by CodeBERT.

```

1 char wf () {
2     Scanner input = new Scanner (System.in);
3     System.out.println ("What is your choice? (x/o)");
4     char choice = input.findInLine (".").charAt (0);
5     while (choice != 'x' && choice != 'o') {
6         System.out.println ("You must enter x or o!");
7         choice = input.next ().charAt (0);
8     }
9     return choice;
10 }

```

Listing 3: Clone29 - A

```

1 char wf () {
2     Scanner input = new Scanner (System.in);
3     System.out.println ("What is your choice? (x/o)");
4     if (input.findInLine (".") != null) {
5         choice = input.findInLine (".").charAt (0);
6         while (choice != 'x' && choice != 'o') {
7             System.out.println ("You must enter x or o!");
8             choice = input.findInLine (".").charAt (0);
9         }
10    }
11    return choice;
12 }

```

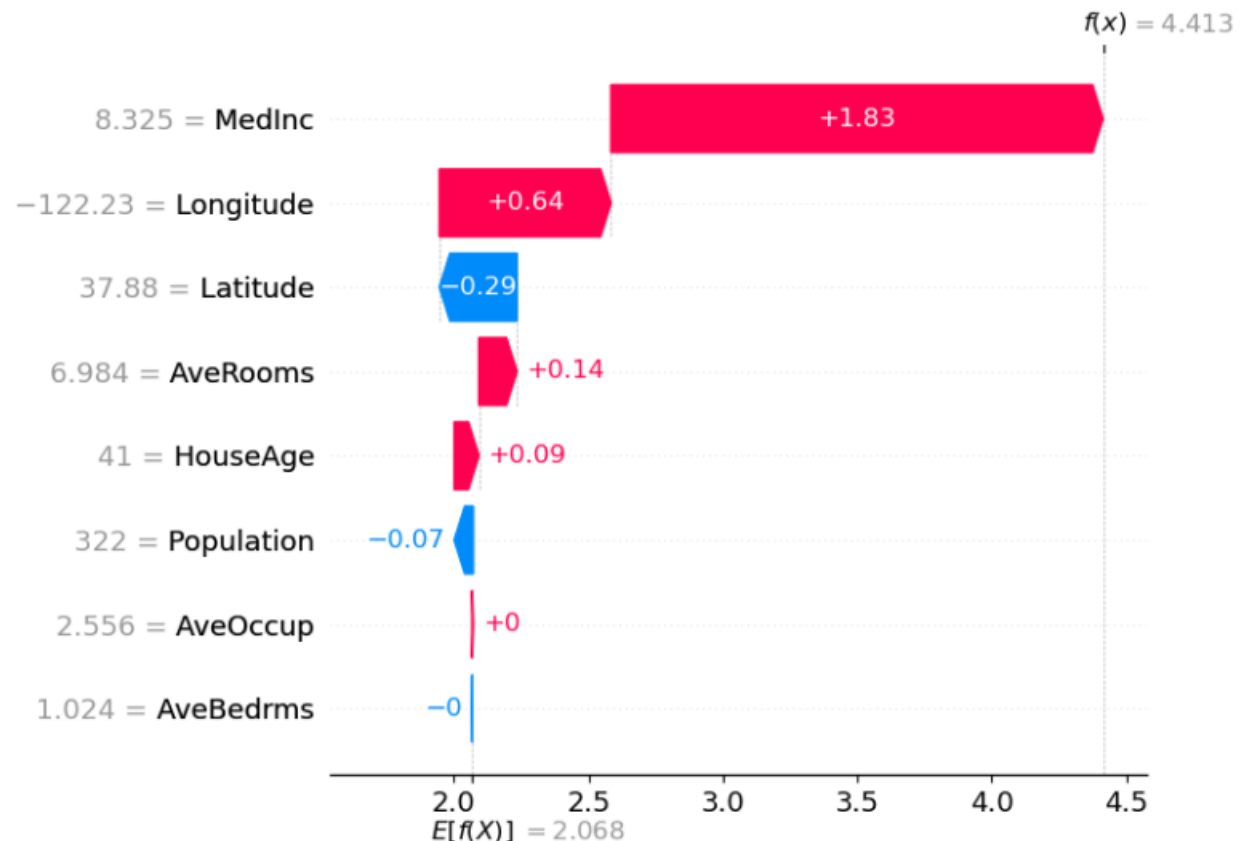
Listing 4: Clone29 - B

SHAP Values

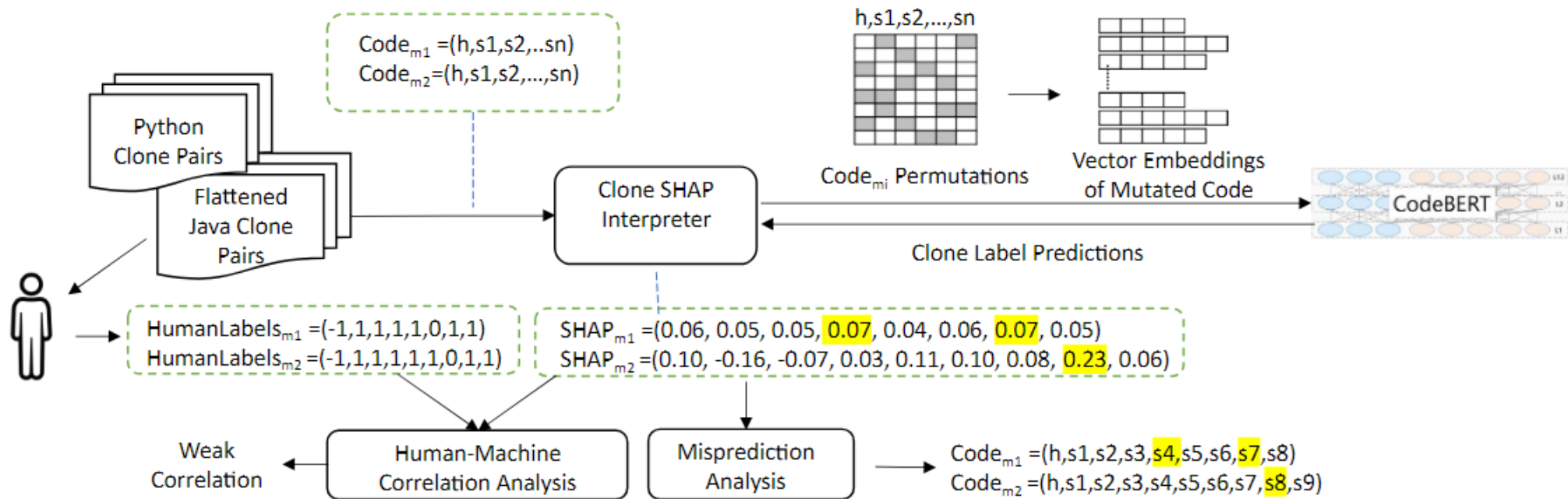
- Spam email
- The word 'free' in the email increased the spamminess of the prediction by 0.2,
- the sender's address 'trustedfriend@email.com' decreased it by 0.1

SHAP Values

- SHAP (SHapley Additive exPlanations)
 - a game theoretic approach to explain the output of any machine learning model.
- Shapley values calculate the importance of a feature by comparing what a model predicts with and without the feature.
- The feature contribution values are between -1 and +1. These values may either be positive or negative.
- Positive SHAP value means positive impact on prediction, leading the model to predict 1.
- Negative SHAP value means negative impact, leading the model to predict 0.



Explaining the predictions of the CodeBERT model for semantic code clone detection

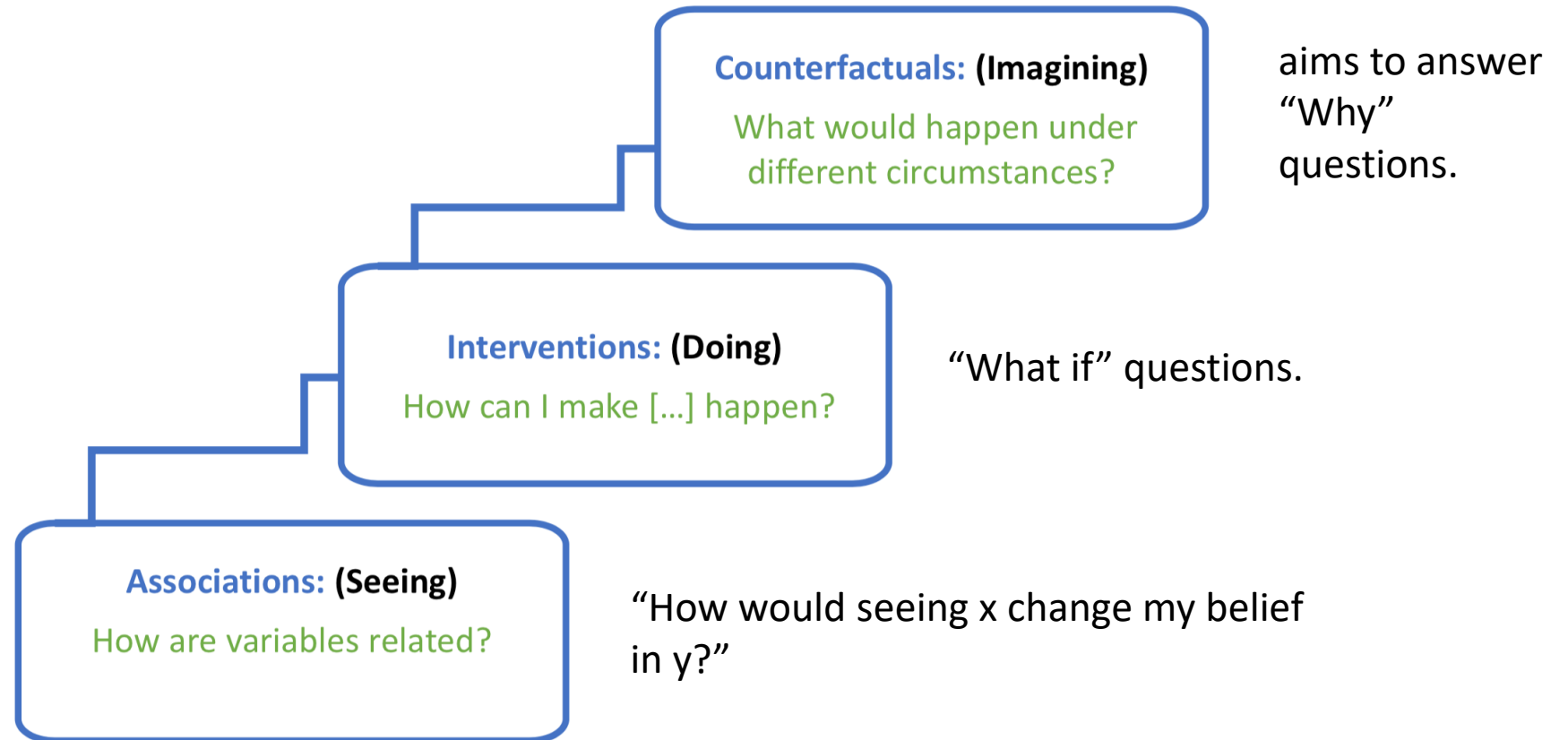
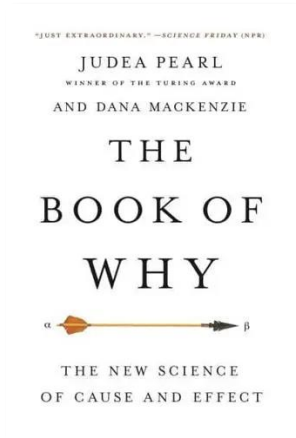


Results

- We use SHAP values to identify key statements of methods in a clone pair that contribute towards a prediction.
 - *Method headers* and *for statements* have the highest average impact on CodeBERT's true predictions and false predictions respectively for Java clone pairs. *Return statements* have the highest average impact on CodeBERT's true and false predictions for Python clone pairs.
- Machine-interpretation of statements correlates with human-intuition only as much as 53.2% of the time for Java and 58.33% of the time for Python clone pairs.
- Thus, we can infer that CodeBERT does not analyze code the same way as humans do. Because of this weak correlation, our confidence in CodeBERT for semantic clone detection is also weak.

Causal Interpretability of Code Models' Decisions for Clone Detection

Ladder of Causality: 3 levels of interpretability



Causal Interpretability

- Causal interpretability
 - helps us understand the *real causes of decisions* made by machine learning algorithms, improve their performance, and prevent them from failing in unexpected circumstances

Causal Interpretability for Clone Detection Models

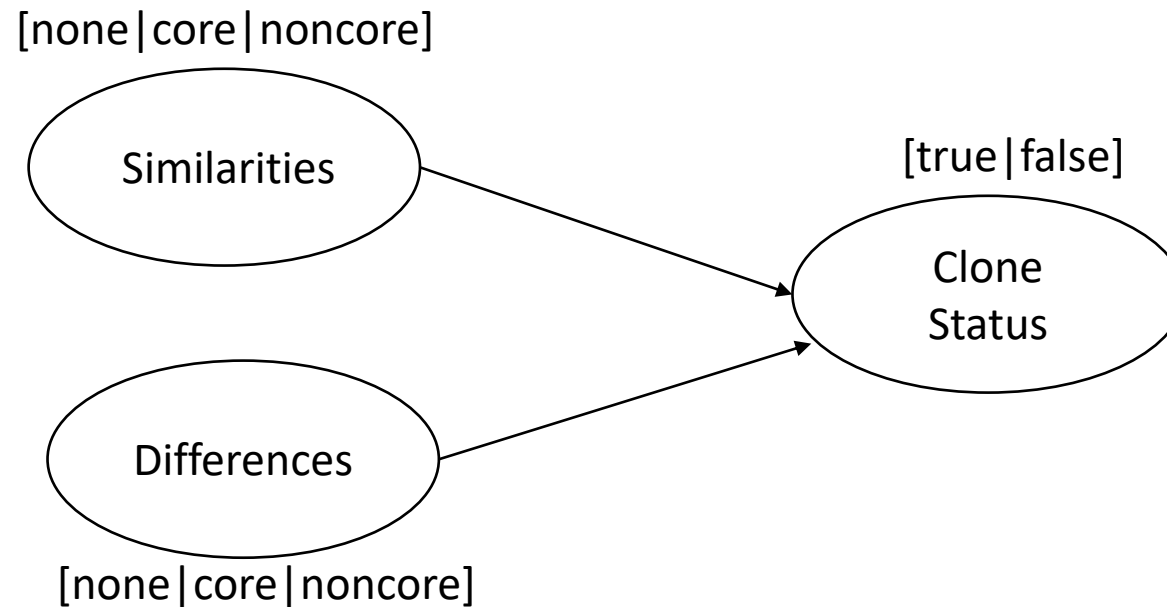
- Was it feature X that caused decision Y ?
 - *“Did the code similarities cause the model to predict the clone as a true clone?”*
- What would have happened to this decision of a classifier had we had a different input to it?
 - *“If we removed the code similarities from a clone pair, would the system still make the same decision?”*
- “Why did the classifier make this decision instead of another?”
 - *Why do we get a false prediction for a clone pair? What’s causing the false prediction?*

Research Goals

- How do we know the real causes of predictions?
 - Are true clone predictions caused by code similarities?
 - Are false clone predictions caused by differences?
 - Are mispredictions caused by distracting similarities or differences?
- How can we decide the best model for clone detection
 - Which is well aligned with human intuition
 - Which is robust, reliable and trustworthy
- How can we measure these attributes?



How to do Causal Inference

1. Causal Diagrams: DAGs used to depict causal relationships between variables, helping to visualize the direction of causality and potential confounding factors.



How to do Causal Inference

2. Counterfactuals: Causal inference often involves comparing observed outcomes with hypothetical outcomes that would have occurred under different conditions or interventions. These hypothetical outcomes are known as counterfactuals.

Observed Outcome	Intervention	Hypothetical Outcome	Hypothetical == Actual Outcome?	Counterfactual Explanation
True clone	Remove similarities	False clone		Similarities are causing the model's original prediction
				The model's original prediction is influenced by confounding factors

How to do Causal Inference

3. Measure Causal Effects: Causal inference quantifies the effect of one variable (the cause or treatment) on another variable (the effect or outcome).

Average Causal Effect Metrics

Causal Interpretation of Code Clone Detection

- Causal framework to interpret a model's clone predictions
 - Are similarities the real cause of clone prediction?
 - Counterfactual explanations help establish causes
 - Using human labels to create counterfactual clone pairs

VisualStudio Annotator Tool for Clone and Code Labeling

```
J Clone13.java
1  public class Clone13 {
2  /*
3  * Semantic clone benchmark
4  * Source code are extracted from Stack Overflow
5  * Stack overflow Question #:453018
6  * Stack Overflow answer #:1647015
7  * And Stack Overflow answer#:39232425
8  */
9  public int countLines (String filename) throws IOException {
10     LineNumberReader reader = new LineNumberReader (new FileReader (filename));
11     int cnt = 0;
12     String lineRead = "";
13     while ((lineRead = reader.readLine ()) != null) {
14     }
15     cnt = reader.getLineNumber ();
16     reader.close ();
17     return cnt;
18 }
19
20 public static int countLines (File input) throws IOException {
21     try (InputStream is = new FileInputStream (input)) {
22         int count = 1;
23         for (int aChar = 0;
24             aChar != - 1; aChar = is.read ()) count += aChar == '\n' ? 1 : 0;
25         return count;
26     }
27 }
28
29 }
```

Label Resolution

- Clone labels
 - Two human annotators and another that breaks ties
- Code labels
 - Two human annotation sets
 - Assign a label value (-2,-1,+1,+2) based on core differences, non-core differences, noncore similarities, and core similarities
 - Calculate average label values for overlapping label segments

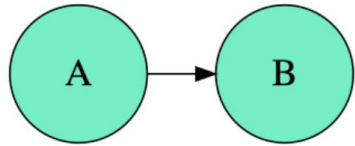
Mutation Strategy

- Syntax-preserving mutations
- Mutation scope
 - Removing only core similarities or differences
 - Removing all core and noncore similarities or differences
- Mutate using AST parser
 - Remove a set of statements
 - Remove single statements
 - Remove parts of a statement

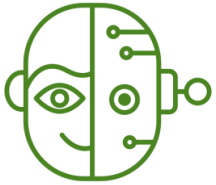
Evaluation



Evaluation Metrics



- Average Causal Effect (ACE) of removing similarities and differences



- Human-model code similarity intuition alignment

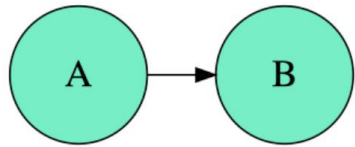


- Confounding Frequency



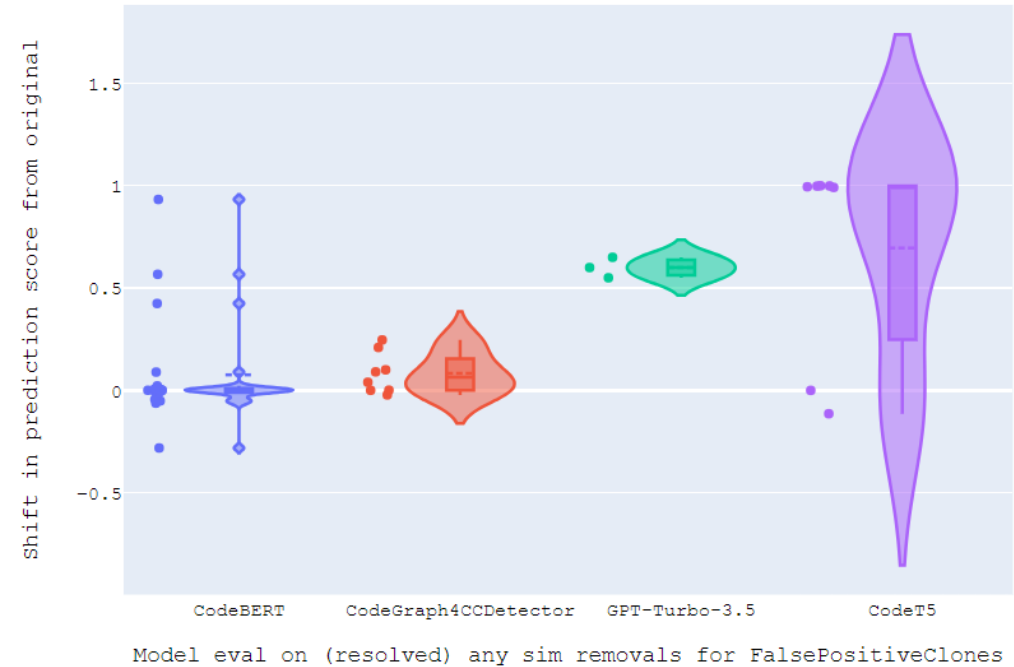
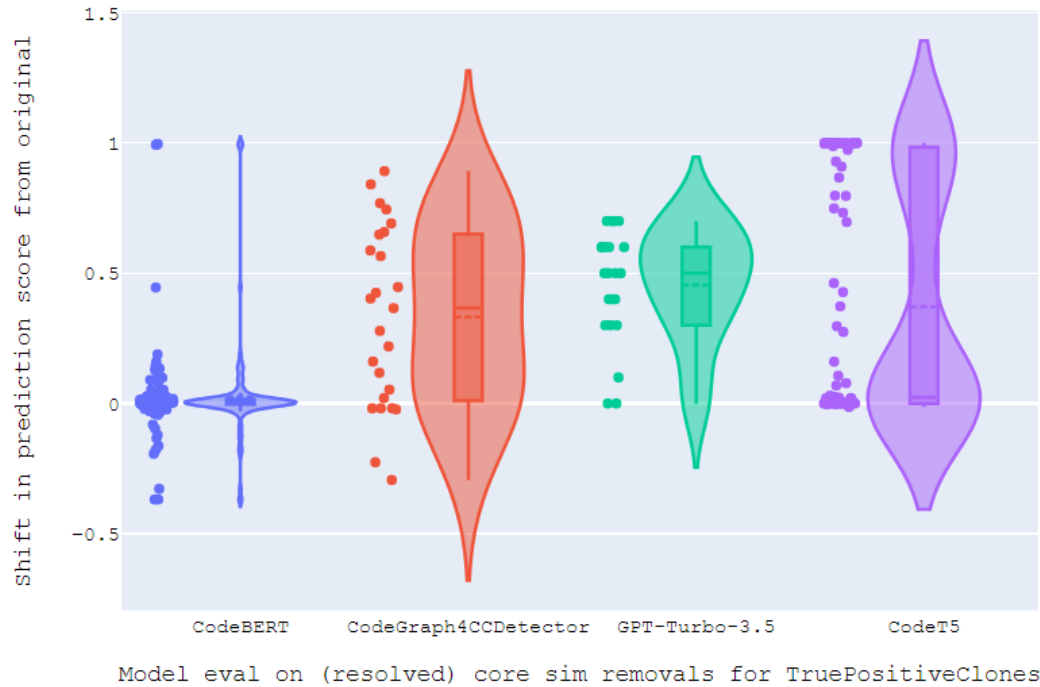
- Prediction Consistency

Evaluation Metrics

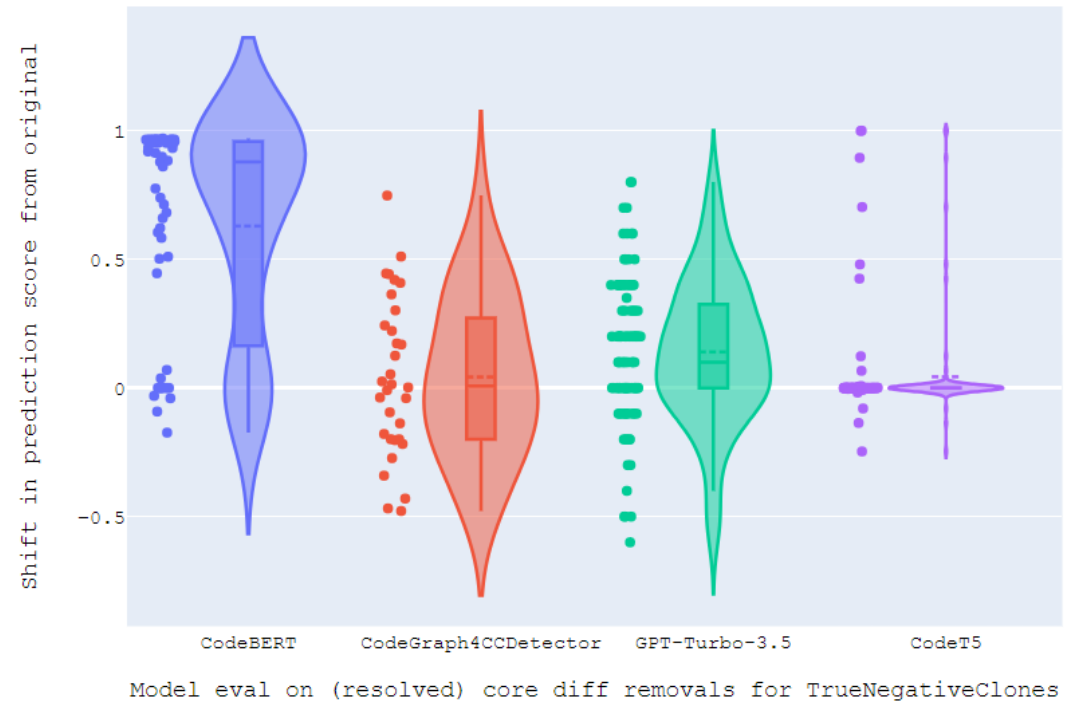
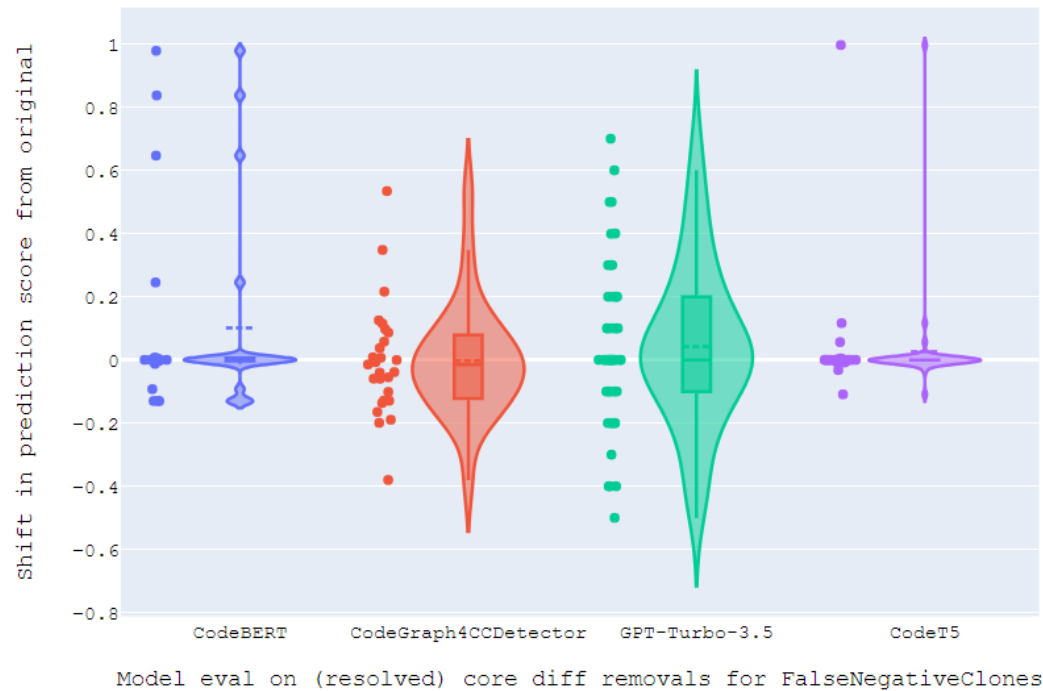


- Average Causal Effect (ACE) of removing similarities and differences
 - Measures the average of the model's prediction shifts on mutated clone pairs
 - ACE of similarities in TP and FP
 - *How much do human-identified similarities influence a model's predictions?*
 - ACE of differences in TN and FN
 - *How much do human-identified differences influence a model's predictions?*
 - A positive causal effect value > 0 means the model aligns with human intuition
 - A 0 or negative causal effect < 0 means the model does not align with human intuition

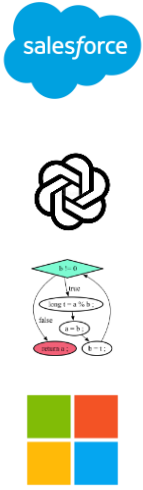

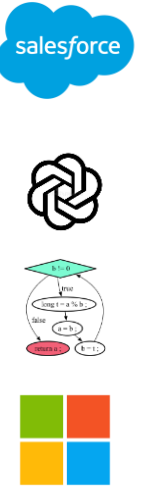


Sensitivity of models' prediction scores to similarities removal

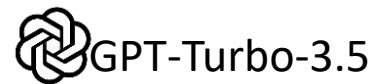


Sensitivity of various models' prediction scores to differences removal

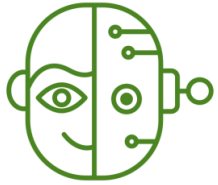


Model Ranks

ACE of sim TP cases	ACE of diff TN cases	ACE of sim FP cases	ACE of diff FN cases	Aggregate ACE
				



Evaluation Metrics



- Human-model code similarity intuition alignment metric
 - Case H=1, M=1.
 - Is model's true clone prediction for a true clone pair based on human-identified code similarities?
 - *Human-model alignment percentage = average no. of prediction flips caused by similarities removal x 100*

CodeBERT	CodeGraph4CCDetector	GPT-Turbo-3.5	CodeT5
4.44%	53.6%	89.03%	49.13%

Evaluation Metrics



- Confounding Frequency Metric

- Measures the number of times a model's *prediction gets flipped* on mutated clone pairs (for FP and FN cases)
- For TP cases, if the *prediction doesn't flip* by removing similarities, we count it as the model being confounded.
- *Confounding frequency = (no. of times flipped on FP + no. of time flipped on FN + no. of times didn't flip on TP) / (|FP+FN+TP|)*
- Model with lower confounding frequency is better

Mutation Scope	CodeBERT	CodeGraph4CCDetector	GPT-Turbo-3.5	CodeT5
Core	0.77	0.2	0.09	0.45
All	0.73	0.2	0.1	0.28

Evaluation Metrics






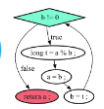

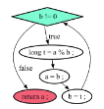

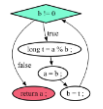








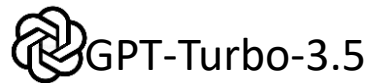
- Prediction Consistency

- A model's predictions across two runs on the same data should be the same
- We calculate the Jaccard similarity between the predictions for a model on the same set of clone pairs






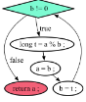

CodeBERT	CodeGraph4CCDetector	GPT-Turbo-3.5	CodeT5
1	1	0.75	1

Model Ranks

F1 Score	Human Alignment for TP sim	Least Confounded	Prediction Consistency
			  
			
			
			



Model Ranks for Semantic Code Clone Detection

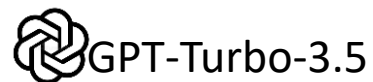
Models	Gold stars
	
	
 	



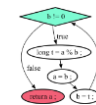
CodeBERT



CodeT5



GPT-Turbo-3.5



CodeGraph4CCDetector

Future work



Using automated techniques to generate the counterfactual samples

Using SOTA model intuitions


- First verify SHAP-based explanations of a SOTA model using human evaluation
- Perform SHAP-based mutations to get counterfactuals
- Evaluate other models on mutated counterfactual samples




Using our labeled data to finetune ML models using contrastive learning

FYI Upcoming Workshop !!!

LLM4Code 2024

 The First International Workshop on Large Language Models for Code

 Co-Located with ICSE 2024

 Lisbon, Portugal

 Apr 16, 2024

 Follow us on Twitter: [@llm4code!](https://twitter.com/llm4code)

Submission deadline: **Dec 7th**, 2023

Thanks!



Online Collaboration and Mentorship

For research consultation:

- Email using subject: (LUMS PhD/MS/BS RESEARCH consultation)

For research volunteer:

- Email using subject: (LUMS BS RESEARCH VOLUNTEER)

For research collaboration:

- Email using subject: (LUMS PhD/MS RESEARCH COLLABORATION)

For general consultation:

- Email using subject: (LUMS PhD/MS/BS GENERAL consultation)



For PhD, research engineer or research scientist position queries at SMU

Email using subject: (LUMS Research Position at SMU)



shamsaabid@smu.edu.sg



<https://shamsa-abid.github.io/>