

# HinglishEval: Evaluating the Effectiveness of Code-generation Models on Hinglish Prompts

Mrigank Pawagi, Anirudh Gupta, Siddharth Reddy Rolla, Kintan Saha

---

Indian Institute of Science, Bengaluru



# Problem

---

- LLMs have been shown to possess strong code-generation abilities<sup>[1]</sup>
- These abilities can be used by students for CS1 courses
- But do these code generation models benefit all students equally?

**Hypothesis : Are LLMs as good at code generation in native languages as compared to English?**

[1] : Majeed Kazemitabaar et al. - "How Novices Use LLM-based Code Generators to Solve CS1 Coding Tasks in a Self-Paced Learning Environment"

# Key Contributions

---

- We **propose a semi-automated technique to translate** English prompts into Hinglish, and we use this technique to create **HinglishEval**: a Hinglish translation of the widely-used code-generation benchmark HumanEval.
- We **compare the performance** of several popular open- and closed source code-generation models on Hinglish and English prompts.

# Let's test this hypothesis for Hinglish

---

## Hinglish:

- A mixture of Hindi and English written in the usual Hindi grammar with English words interspersed in between.

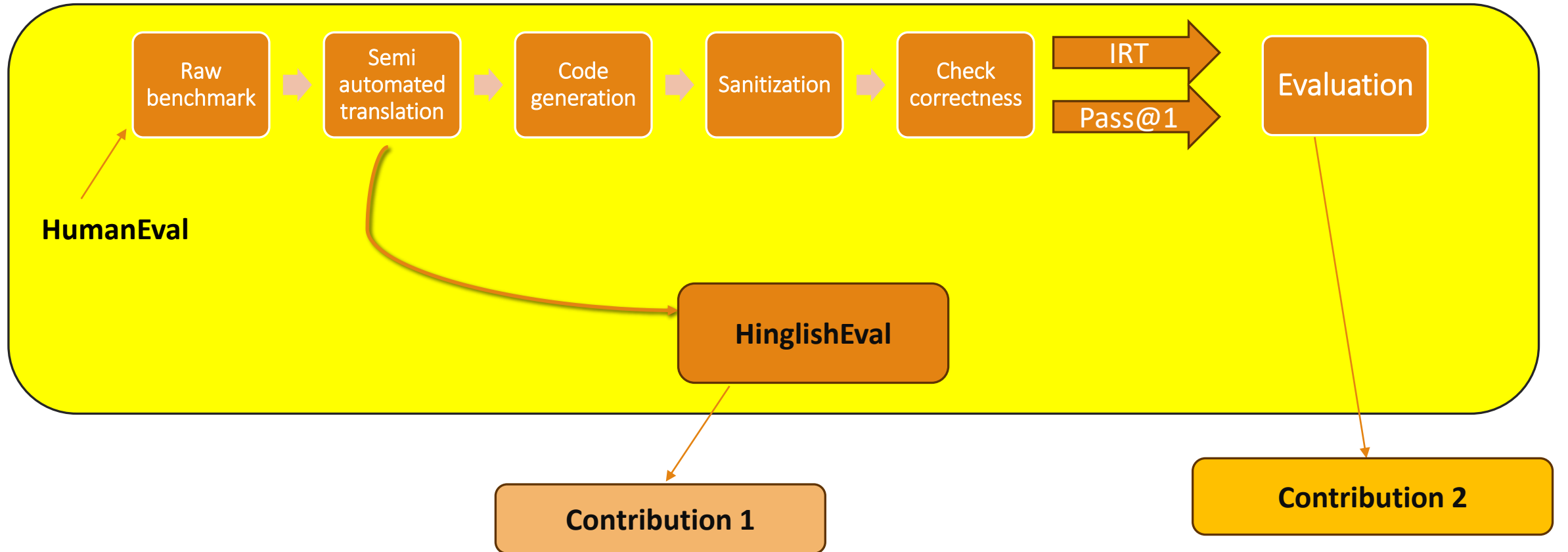
*"Write a code to generate the first n Fibonacci numbers in Python."*



*"Pehle n Fibonacci numbers ko generate karne ke liye Python mei code likho."*

- To test our hypothesis on Hinglish we need a dataset containing standardised introductory programming questions in Hinglish.
- Hence, we propose to create a new benchmark to evaluate models on this task.

# PIPELINE OF HINGLISHEVAL



# Creating the Benchmark

## Semi-automated translation

HumanEval (164 Problems)



Automatic translation



Manual Verification



HinglishEval

**System:** You are a translator fluent in both Hindi and English. Today, you will convert docstrings of Python functions from English to Hinglish, which is a conversational form of Hindi in which we use English for technical words related to syntax or code, programming concepts, and mathematics. Note that all text must be in the Roman script, like in the example.

**User:**

Given a positive integer `n`, return the product of the odd digits.  
Return 0 if all digits are even. For example:

```
digits(1) == 1
digits(4) == 0
digits(235) == 15
```

**Assistant:**

Diye gaye positive integer `n` ke odd digits ka product return karo.  
Agar saare digits even ho to 0 return karo. Jaise ki:

```
digits(1) == 1
digits(4) == 0
digits(235) == 15
```

**User:** <docstring>

Write a function that accepts a list of strings.  
The list contains different words. Return the word with maximum number of unique characters. If multiple strings have maximum number of unique characters, return the one which comes first in lexicographical order.  
...



Ek function likho jo strings ki **ek list** accept karta hai.  
List mein alag alag **words** hain. **Sabse zyada unique characters** wala **word** return karo. Agar multiple strings mein maximum number of unique characters ho, toh lexicographical order mein sabse pehle aane wala **word** return karo.  
...



Ek function likho jo strings ki **list ko** accept karta hai.  
List mein alag alag **shabd hote** hain. **Unique characters ki maximum number** wala **shabd** return karo. Agar multiple strings mein maximum number of unique characters ho, toh lexicographical order mein sabse pehle aane wala **shabd** return karo.  
...

# Manual Verifications

Converting the translations into conversational Hinglish

# HinglishEval

Translated dataset is available as HinglishEval.json

A screenshot of a GitHub repository's file browser. The repository is named 'Final Refactors (#73)' and is on the 'main' branch. The file list includes folders like '.github/workflows', 'irt', 'samples', and 'translation', and files like '.gitignore', 'HinglishEval.json', 'README.md', 'example.env', and 'requirements.txt'. The file 'HinglishEval.json' is circled in orange. The commit history for 'HinglishEval.json' shows a change by 'Final Refactors (#59)' 10 months ago.

File/Folder	Commit	Time
.github/workflows	flake8.yml: add irt/ and samples/ (part-2)	3 days ago
irt	Final Refactors (#73)	3 days ago
samples	Final Refactors (#73)	3 days ago
translation	Final Refactors (#73)	3 days ago
.gitignore	results	9 months ago
HinglishEval.json	changes in HinglishEval.json (#59)	10 months ago
README.md	Final Refactors (#73)	3 days ago
example.env	Final Refactors (#73)	3 days ago
requirements.txt	Final Refactors (#73)	3 days ago

<https://tinyurl.com/hinglisheval>



# HinglishEval Problem 0

## Prompt

```
from typing import List

def has_close_elements(numbers: List[float], threshold: float) -> bool:
    """
    Diye gaye numbers ki list mein, kya koi do numbers ek dusre se diye
    gaye threshold se zyada close hain.
    >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
    False
    >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3)
    True
    """
```

## GroundTruth Solutions

```
for idx, elem in enumerate(numbers):
    for idx2, elem2 in enumerate(numbers):
        if idx != idx2:
            distance = abs(elem - elem2)
            if distance < threshold:
                return True

return False
```

## Test Cases

```
def check(candidate):
    assert candidate([1.0, 2.0, 3.9, 4.0, 5.0, 2.2], 0.3) == True
    assert candidate([1.0, 2.0, 3.9, 4.0, 5.0, 2.2], 0.05) == False
    assert candidate([1.0, 2.0, 5.9, 4.0, 5.0], 0.95) == True
    assert candidate([1.0, 2.0, 5.9, 4.0, 5.0], 0.8) == False
    assert candidate([1.0, 2.0, 3.0, 4.0, 5.0, 2.0], 0.1) == True
    assert candidate([1.1, 2.2, 3.1, 4.1, 5.1], 1.0) == True
    assert candidate([1.1, 2.2, 3.1, 4.1, 5.1], 0.5) == False
```

# Generating Python Codes For HinglishEval

---



## LOCALLY RUN MODELS

- Code generated by running models locally, downloaded from HuggingFace.
- Models under this category include Gemma, Codegen, Polycoder and their versions.
- No System Prompt Provided

The quality of models without the System Prompt is worse than with System Prompts.



## MODELS RUN via API

- Code generated through API (using OpenAI & DeepInfra)
- Models under this category include Gpt, Llama, MistralAI, Phi-3-medium and their versions.
- Common System Prompt for these Models

You are an experienced Python programmer. Complete the Python functions from the given docstrings. Do NOT write anything except the function definition. Avoid print and input statements.

# Contribution : Evaluation of various code-generation models

---

To assess the effectiveness of code-generation by the LLMs, we adopt quantitative evaluation techniques such as :

1. **Pass@k** - A metric that measures the likelihood of generating a correct code in the first 'k' attempts. (**Note** : k=1 in our evaluation).
2. **Item Response Theory (IRT)** - A statistical framework that evaluates model performance based on individual difficulties and discriminations of the problems.

These ratings give us an idea about the performance of the various models under test. This comparative study also gives us an indication of their efficacy when prompted in a different language.

# Pass @ k

---

The '**Pass@k**' is a standard metric introduced by the HumanEval dataset to evaluate the functional correctness of various generated code samples.

It is defined as the probability that atleast one of the top-k generated code samples pass all the unit tests.

Formally :

$$pass@k := \mathbb{E}_{\text{problems}} \left[ 1 - \frac{C(n - c, k)}{C(n, k)} \right]$$

- $c$  = number of code samples which pass all the unit tests.
- $n$  = Total number of code samples.

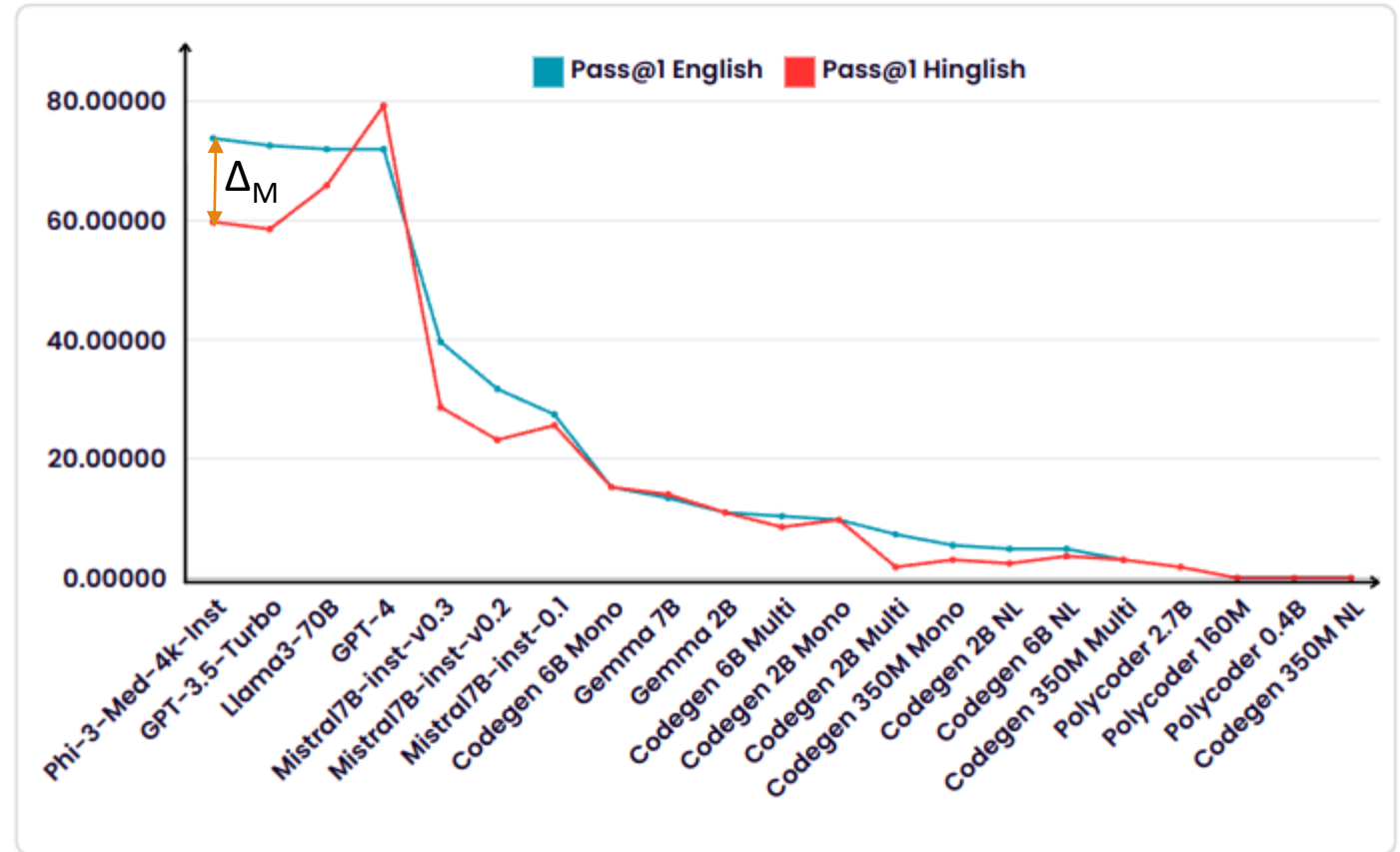
**Note** : Throughout this paper, we use  $k=1$  ie. Pass@1 scores. Thus, the formula simplifies into

$$Pass@1 := \frac{c}{n}$$

# Pass@1 Scores

## Observations:

- The **trend** between models on English and Hinglish prompts seems to be almost **similar**.
- The closed source models in general perform much better on both kinds of prompts.
- The **pass@1** for **English** prompts is generally **higher than** for **Hinglish** prompts but the difference is not very pronounced.
- LLMs with more parameters generally have a better performance.
- **Exceptional behaviour:** GPT-4 and Gemma 7B — negative values of  $\Delta_M$



- $P_1(M,E)$  = pass@1 score of model for English prompts
- $P_1(M,H)$  = pass@1 score of model for Hinglish prompts
- $\Delta_M = P_1(M,E) - P_1(M,H)$

# Item Response Theory

---

- **Statistical framework** to evaluate model performances.
- Dependent on a **logistical distribution curve**.
- Accounts for **varying nature** of questions.

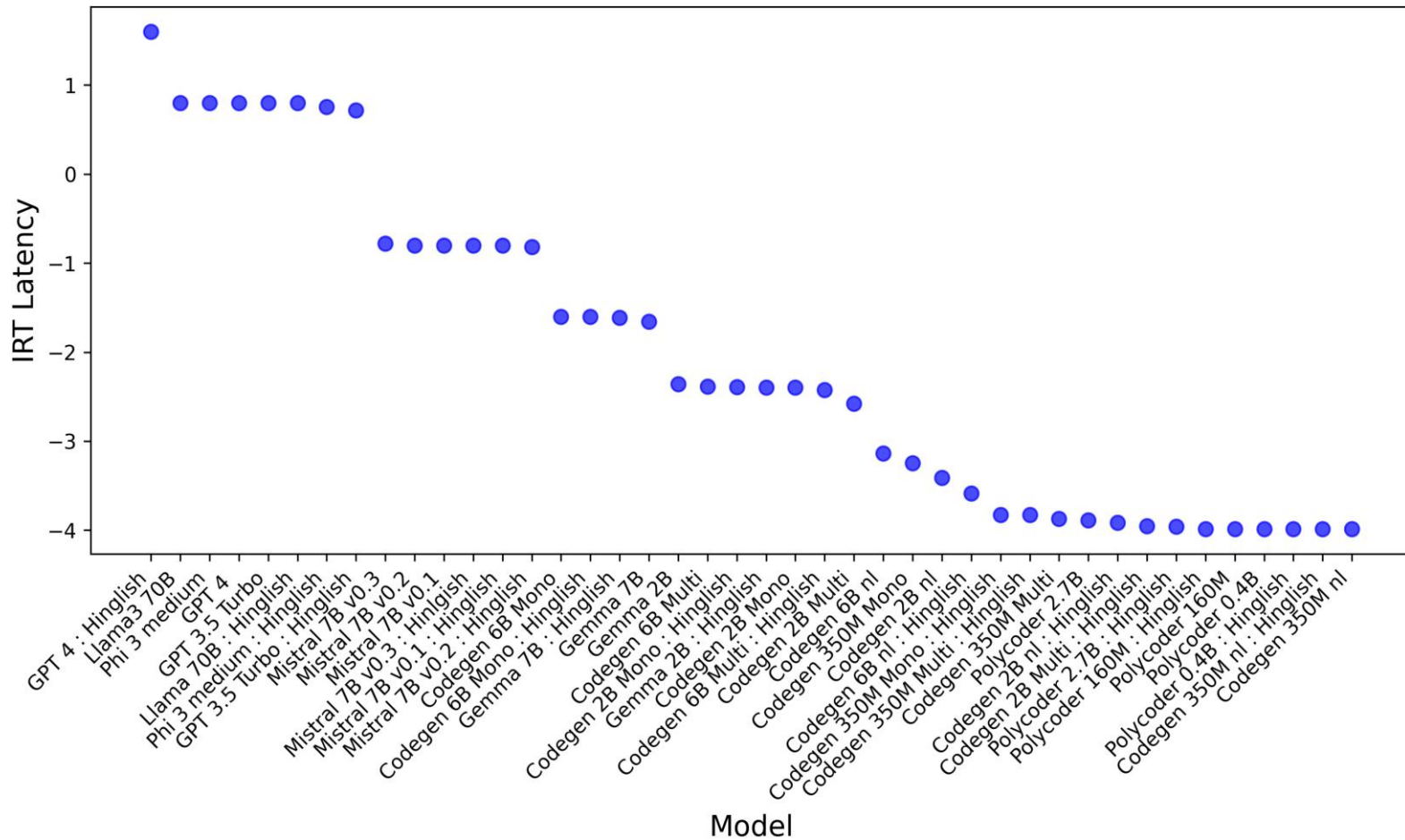
We use the 2-Parameter Logistic IRT model for our analysis. It consists of the following two parameters on which the model's latent ability depends :

- **Difficulty** (b) : Measure of how challenging the problem is.
- **Discrimination** (a) : Measure of how well a test case differentiates between high and low performing models.

$$P(\theta) = \frac{1}{1 + e^{-a(\theta-b)}}$$

# Latency

- Measure of ability of model's answering capabilities.
- It is a function of the model's responses by considering the difficulty distributions of the questions.



A single combined IRT evaluation is used due to :

- Our assumption/ hypothesis that the difficulties of various problems should be **independent** of the language they are presented in.
- The **translation process** did not in any way add new or remove any information from the question.

# Some interesting case studies

## Example 1

```
1 def match_parens(lst):
2     """
3     Aapko do strings ki ek list di gayi hai, dono strings me sirf open
4     ↪ parentheses '(' ya close parentheses ')' hain. Aapka kaam ye
5     ↪ check karna hai ki kya in dono strings ko kisi order me
6     ↪ concatenate karne se resulting string acchi banegi. Ek string S
7     ↪ sirf aur sirf tabhi acchi maani jayegi agar usme saare
8     ↪ parentheses balanced ho...
9     """
10    def is_balanced(s):
11        ...
12    return 'Yes' if is_balanced(lst[0] + lst[1]) else 'No'
```

HumanEval Problem 119

Mistral-7B-v0.3's response for the Hinglish version. The bug in this response can be fixed by changing the high- lighted text to *is\_balanced(lst[0] + lst[1])* or *is\_balanced(lst[1] + lst[0])*.

**LOGICAL ERROR**



# Example 2

```
1 def generate_integers(a, b):  
2     """  
3     Do positive integers a aur b diye gaye hain, a aur b ke beech ke  
4     ↪ even digits ko ascending order mein return karo...  
5     """  
6     return [i for i in range(min(a, b), max(a, b) + 1) if i % 2 == 0]
```

HumanEval Problem 163

GPT-4's response for the Hinglish version. The bug in this response can be fixed by changing the highlighted text to `i % 2 == 0 and 0 <= i <= 9`.

**LOGICAL ERROR!**

# Example 3

---

```
1 def words_string(s):
2     """
3     Aapko ek string di jayegi jisme words commas ya spaces se separated
4     ↪ honge. Aapka task hai string ko words mein split karna aur words
5     ↪ ki array return karna...
6     """
7     words = s.split(', ')
8     words = [word.strip() for word in words]
9     return words
```

HumanEval Problem 101

GPT-3.5-Turbo's response for the Hinglish version. The bug in this response can be fixed by changing the highlighted text to `s.replace(' , ', ' ').split()`.

**LOGICAL ERROR!**

# Other common errors by LLMs

---

There are other mistakes by models apart from Logical Errors which include:

- 1) Hallucination
- 2) Syntax Errors
- 3) Missing imports

```
1 def right_angle_triangle(a, b, c):
2     '''
3     Given the lengths of the three sides of a triangle. Return True if the three
4     sides form a right-angled triangle, False otherwise.
5     A right-angled triangle is a triangle in which one angle is right angle or
6     90 degree.
7     Example:
8     right_angle_triangle(3, 4, 5) == True
9     right_angle_triangle(1, 2, 3) == False
10    '''
11    a = math.radians(a)
12    b = math.radians(b)
13    c = math.radians(c)
14    for i in [a, b, c]:
15        if math.pow(math.sin(i/2), 2) + math.cos(i) * math.cos(i + math.radians(90)) >= 0:
16            return True
17    return False
```

# Future Work

---

- Extrapolation of this benchmark to various other languages.
- Improving the scope of the problems covered.
- Extending it to a more difficult set of problems to include intermediate and advanced learners.
- A more comprehensive benchmark covering more aspects of software engineering such as debugging, code comprehension etc.
- Conducting practical studies to test the validity.

**Thank  
You**

**Questions?**



# In case you wish to collaborate or contribute :

- Siddharth Reddy : [siddharthrr@iisc.ac.in](mailto:siddharthrr@iisc.ac.in)
- Anirudh Gupta : [anirudhgupta@iisc.ac.in](mailto:anirudhgupta@iisc.ac.in)
- Kintan Saha : [kintansaha@iisc.ac.in](mailto:kintansaha@iisc.ac.in)
- Mrigank Pawagi : [mrigankp@iisc.ac.in](mailto:mrigankp@iisc.ac.in)