Data Hound: Linking Educational Value to LLM Code Completion Performance During Inference

EEMCS, Delft University of Technology

Author: Boris Annink <B.R.M.Annink@student.tudelft.nl>
Supervisors: Jonathan Katzy and Razvan Popescu

```
def f(x):
  return 1 if x==0 else x*f(x-1);
print(f(5))
```

```
def factorial(n):
    """
    Calculate the factorial using recursion.
    Args:
        n (int): A non-negative integer.
    Returns:
        int: The factorial of n.
    """
    if n == 0:
        return 1
    return n * factorial(n - 1)

# Example use
print(factorial(5))  # Output: 120
```

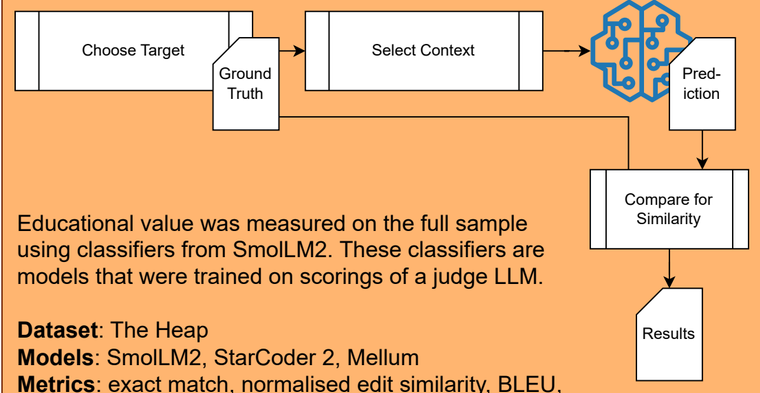
Background

LLMs work by continuously inferring (i.e. predicting) the next token of the input. Model size is an important factor as larger models are capable of better understanding natural language.

Data smells are patterns in data that are not desired in the dataset. Removing smells from the training dataset prevents models from learning the undesired patterns, increasing performance.

Educational value is a quality of text that scores how suitable it is for education. This research focuses on the educational value of code, where high educational code is defined as valid, practical, complex (i.e., advanced programming topics), self-contained, and educationally worded.

Methodology



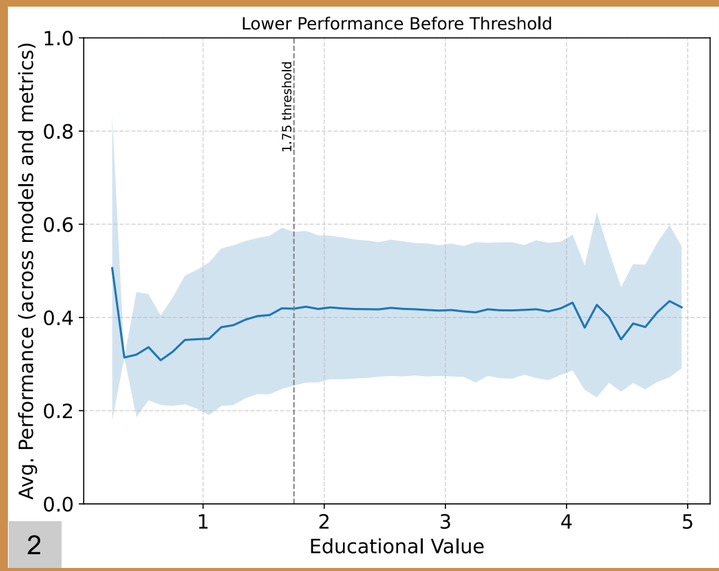
Educational value was measured on the full sample using classifiers from SmolLM2. These classifiers are models that were trained on scorings of a judge LLM.

Dataset: The Heap
Models: SmolLM2, StarCoder 2, Mellum
Metrics: exact match, normalised edit similarity, BLEU, METEOR, ROUGE-L

Research Aim

Recent models have shown improvements in performance after filtering low educational value code from their training dataset. This research investigates whether educational value is also relevant during the inference step in LLMs. It aims to answer the following research questions:

RQ1: What is the distribution of educational value in The Heap dataset?
RQ2: What is the impact of educational value on code completion performance of large language models during inference?
RQ3: How do task type, programming language, model size, and educational value granularity influence this impact?



Conclusion

As seen in Figure 1, the distribution of educational values in the heap is roughly normal (mean ≈ 2.25, SD ≈ 0.45). Samples with extreme educational values are rare.

The results of Figure 2 indicate the existence of a turning point, as significantly lower performance is observed before 1.75. A possible reason could be that the model predicts higher quality code, causing a mismatch. This is supported by the observation in Figure 6, where the line educational value performs the worst for that range.

Figures 3–5 reveal that model size, language, and task type affect performance, though only task type shows interaction with educational value. This difference may stem from how the tasks are framed: next-line prediction treats input as text, whereas next-20-token operates at the token level.

These findings can be used practically for both future models and testing datasets. For example, LLMs in IDEs could be prevented from inferring on low educational code, limiting bad suggestions and saving computing time. More importantly, it can be used for enhancing testing datasets to either highlight or ignore low educational code.