

On using an LLM-based Approach for Automatic Evaluation of Programming Tasks in MOOC Videos

Corinna Kindlhofer, Josef Wachtler, Martin Ebner

Educational Technology, Graz University of Technology
Münzgrabenstraße 36/I, 8010 Graz, Austria - Europe

Contents

- Introduction, Motivation
- Implementation
- Evaluation
- Conclusion

Introduction

Background and Motivation

- Interactive videos increase learner engagement
- Transform passive activity of watching videos into active learning process
- **LIVE**: Web platform that integrates interactive tasks into videos
 - E.g., multiple-choice questions and programming tasks
- Limitation: No automatic evaluation of programming tasks and no automatic feedback for code submissions



Logo source: <https://live.learninglab.tugraz.at/static/images/live-cloud-v02.png>

Research Questions

Can a self-hosted LLM be used to automatically evaluate small programming tasks integrated into MOOC videos?

Is the feedback generated by a self-hosted LLM appropriate and beneficial for students in MOOCs in comparison to semiautomatic feedback?

Implementation

LLM Selection Process

- Requirements for local execution on a small webserver:
 - Accurate evaluation of code correctness including brief feedback
 - Maximum model size of 5 GB (ideally smaller)
 - Inference time ≤ 400 ms
 - German language support
 - Non-proprietary license
 - Local execution
 - Minimal RAM usage

LLM Selection Process

- Potential models were tested both manually and automatically by running predefined test cases
- Variant of *Microsoft Phi-3.5-mini* was chosen
- Optimization of the performance using different parameters and different prompt variations

LLM Selection Process

- Potential models were tested both manually and automatically by running predefined test cases
- Variant of *Microsoft Phi-3.5-mini* was chosen
- Optimization of the performance using different parameters and different prompt variations

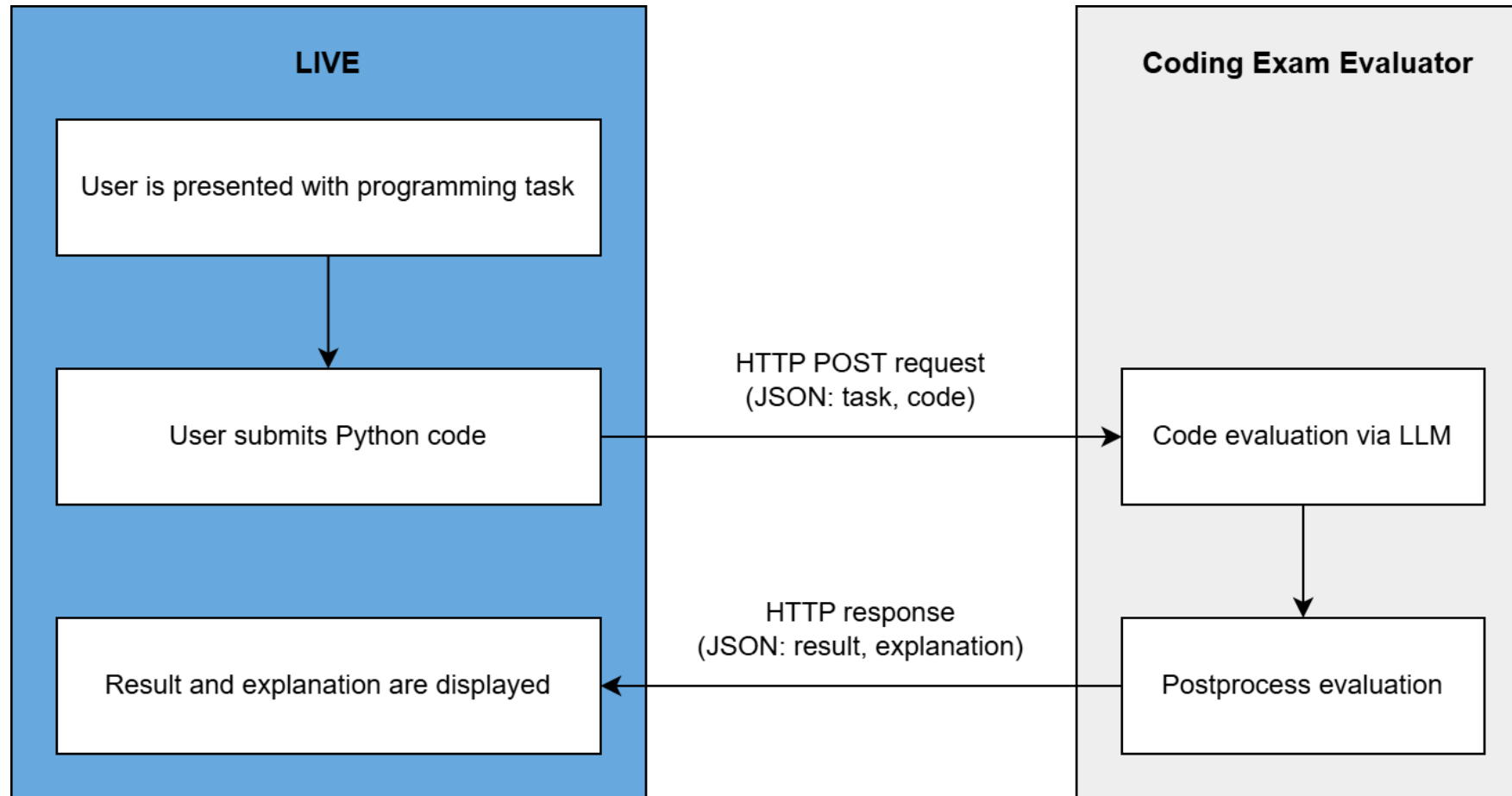


Inference time sometimes higher than defined initially

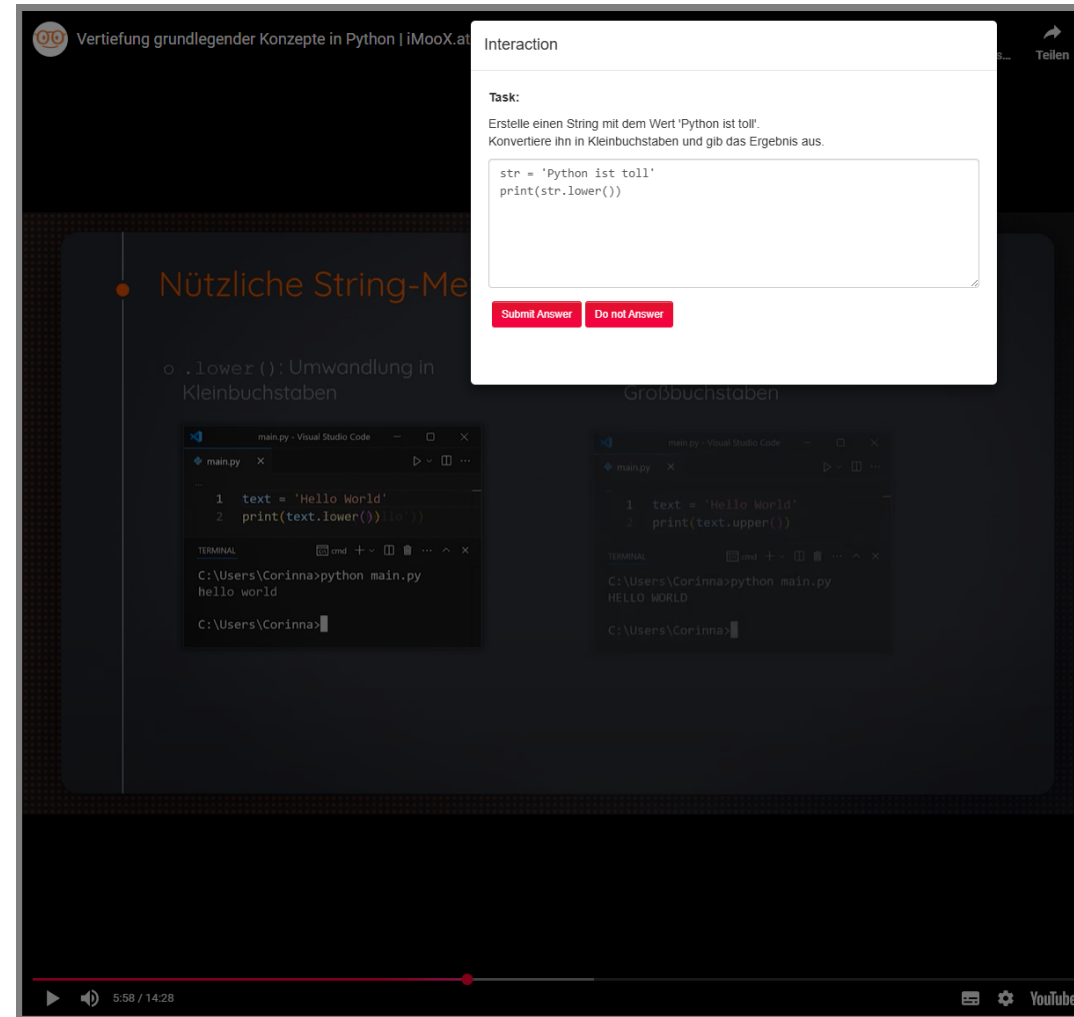
Implementation Details

- Implementation of a stand-alone application ("Coding Exam Evaluator") using an LLM for code evaluation
- Integration into existing LIVE web application
- Teachers can manually correct results if needed

Overview



Screenshots



Screenshots

Interaction

Task:

Erstelle einen String mit dem Wert 'Python ist toll'. Konvertiere ihn in Kleinbuchstaben und gib das Ergebnis aus.

Teacher answer:

```
string = 'Python ist toll'
print(string.lower())
```

Your answer:

```
str = 'Python ist toll'
print(str.lower())
```

Correct output:

```
python ist toll
```

Your output:

```
python ist toll
```

Your Output is correct

Result:

Well done! You solved the task correctly.

OK

Interaction

Task:

Erstelle einen String mit dem Wert 'Python ist toll'. Konvertiere ihn in Kleinbuchstaben und gib das Ergebnis aus.

Teacher answer:

```
string = 'Python ist toll'
print(string.lower())
```

Your answer:

```
str = 'Python ist toll'
print(str.upper())
```

Correct output:

```
python ist toll
```

Your output:

```
PYTHON IST TOLL
```

Your Output is not correct

Result:

Unfortunately, the task was not solved correctly. Possible explanation for the error:

```
Konvertierung in Kleinbuchstaben fehlt
```

OK

Evaluation

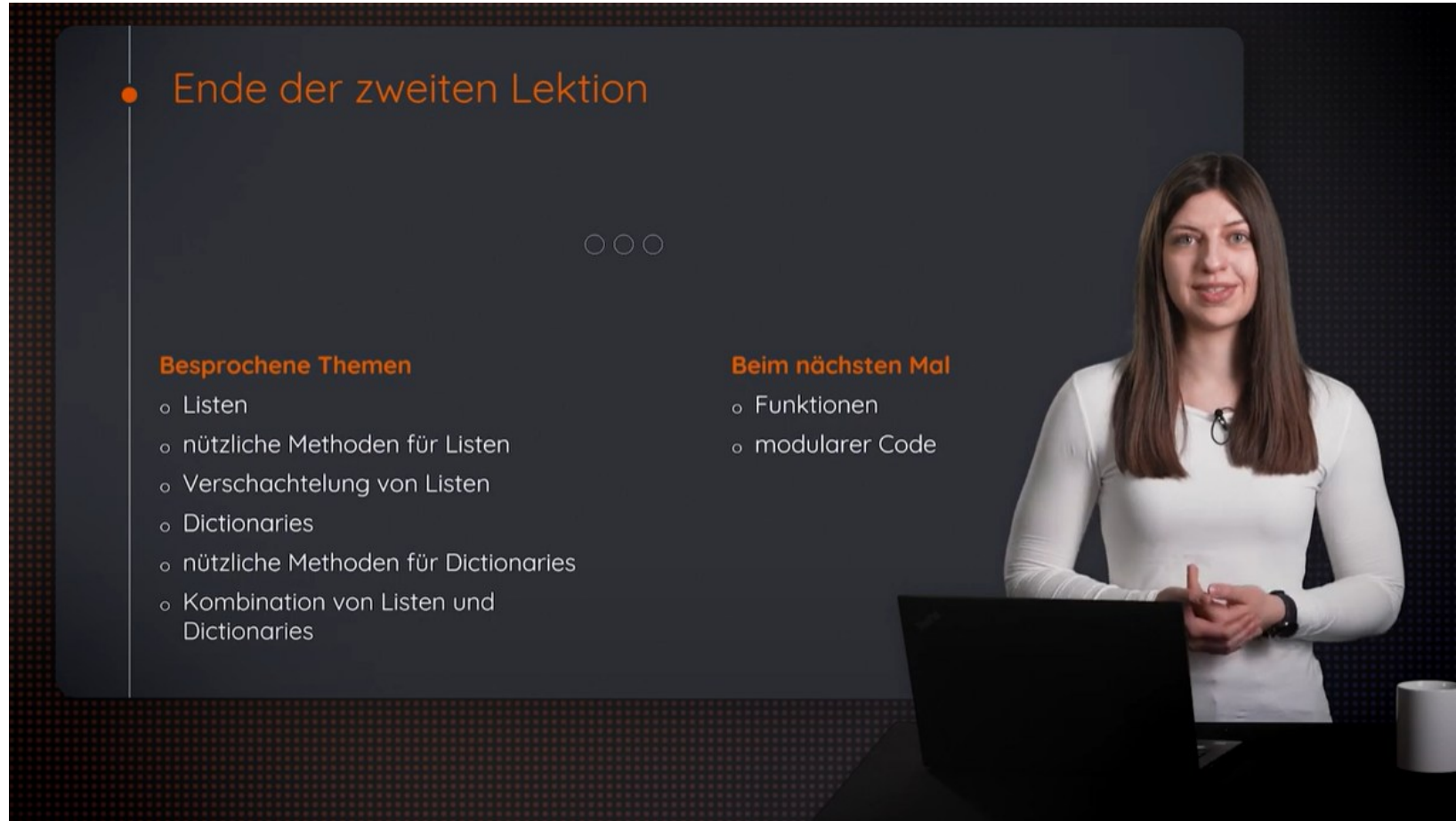
MOOC – Learn Programming with Python: Additional concepts

- Part of a series for participants with no or basic programming knowledge
- Hosted on iMooX.at (in German)
- 4 units, covering topics like data structures, functions and modular code
- Includes videos supported with interactive programming tasks, supplementary materials, a self-assessment, and quizzes



Logo source: <https://imoox.at/mooc/theme/imx/pix/logo.svg>

MOOC – Learn Programming with Python: Additional concepts



• Ende der zweiten Lektion

○○○

Besprochene Themen

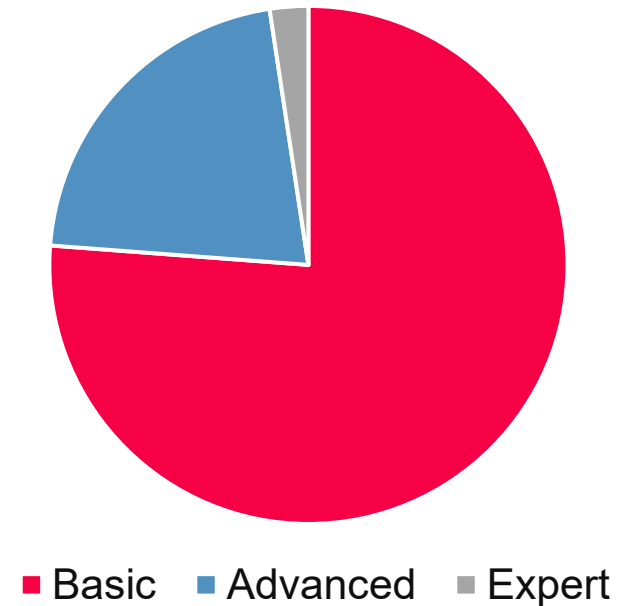
- Listen
- nützliche Methoden für Listen
- Verschachtelung von Listen
- Dictionaries
- nützliche Methoden für Dictionaries
- Kombination von Listen und Dictionaries

Beim nächsten Mal

- Funktionen
- modularer Code

User Statistics

- **Total participants:** 309 (enrolled in the iMooX course as of June 2025)
- **Self-assessment participants:** 42 ($\approx 13.6\%$)
- **Python knowledge (self-rated):**
 - $\approx 76.2\%$ Basic
 - $\approx 21.4\%$ Advanced
 - $\approx 2.4\%$ Expert



Quizzes

- Each unit included an initial and final quiz with identical questions
- Used to measure student's learning progress within a unit
- Maximum score of 10 points per quiz
- Average scores improved across all units from initial to final quizzes

Programming Tasks

- 6 programming tasks across all units
- 143 total attempts across all tasks
 - Evaluate LLM performance by analyzing the generated response
 - Out of these attempts, 88 ($\approx 61.5\%$) required no manual correction of the LLM's response, while 55 ($\approx 38.5\%$) did

Programming Tasks

- Manual correction categories:
 - Alternative implementation
 - Imprecise string matching
 - Incorrect explanation
 - Missing logic not detected
 - Poor language quality
 - Undetected syntax error

Model Comparison – Self-hosted LLM vs. *GPT-4o*

- Incorrectly evaluated tasks were re-tested with *GPT-4o*
- *GPT-4o* performed better in 5 out of 6 categories
- Still failed to detect errors in category **Alternative implementation**

Model Comparison – Self-hosted LLM vs. *GPT-4o*

- Incorrectly evaluated tasks were re-tested with *GPT-4o*
- *GPT-4o* performed better in 5 out of 6 categories
- Still failed to detect errors in category **Alternative implementation**

 However: *GPT-4o* is much larger, proprietary and cannot be run locally

Comparison with Non-LLM-based Evaluation of Programming Tasks

- **Different MOOCs and different participants**
- Higher average improvement between initial and final quizzes: 1.93 points (vs. 1.51 points in previous implementation)
- In some units, the feedback may have had a greater impact, whereas in others the improvements were not significant

Conclusion

Conclusion

- Successfully developed an automated feedback system for simple Python programming tasks using an LLM
- Evaluation was conducted within a Python programming MOOC
- Manual corrections were still required for some programming task submissions
- Feedback was largely found to be appropriate and beneficial for students
- Combining automated and manual evaluation seems to be the most effective approach for now

Thank you ;-)

- Questions?
- Educational Technology, Graz University of Technology
Münzgrabenstraße 36/I, 8010 Graz, Austria
- <https://elearningblog.tugraz.at/>
- <http://www.josefwachtler.at/>