Tic-Tac-Toe mit Llama3.2

Tumar Chegebaeva

Multimodale Mensch Computer Interaktion

July 14, 2025

Übersicht

1. Implementierung

2. Ergebnisse

3. Zusammenfassung

Implementierung

Einfaches Tic-Tac-Toe

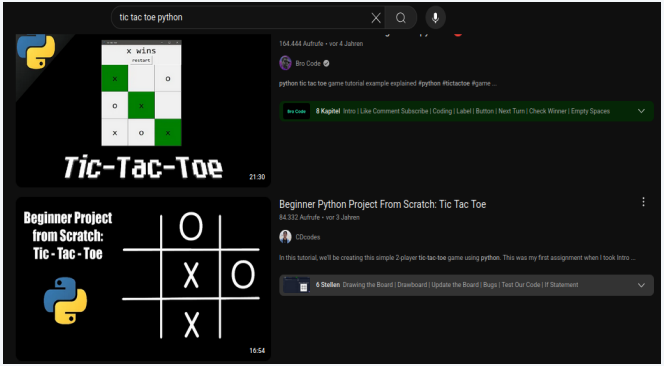


Figure: Grundgerüst nach Video-Tutorials

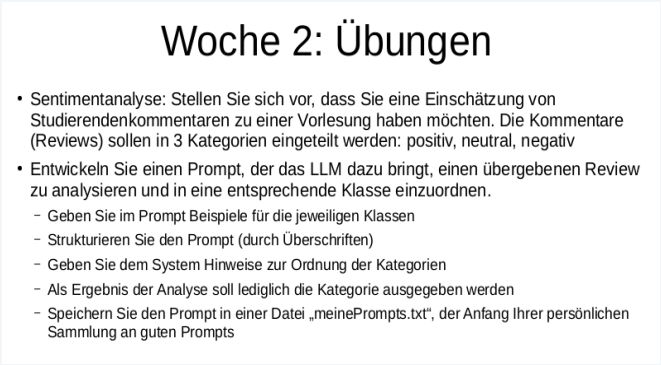


Figure: Erste Überlegung zum Prompt-Design

Anbindung des Sprachmodells

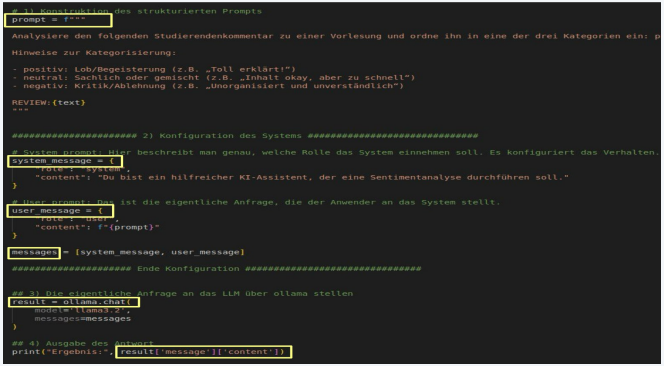


Figure: Beispielhafte Prompt-Konfiguration

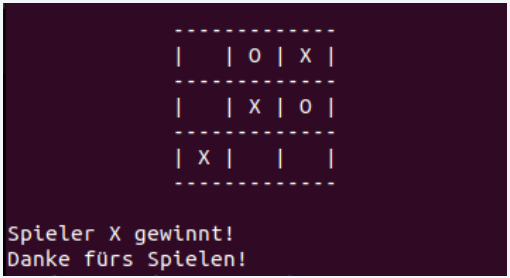
```python
import os
import ollama

board = [' '] * 9

def display_board():
    print(f"""
             -------------
             | {board[0]} | {board[1]} | {board[2]} |
             -------------
             | {board[3]} | {board[4]} | {board[5]} |
             -------------
             | {board[6]} | {board[7]} | {board[8]} |
             -------------
        """)

def check_winner():
    wins = [
        [0, 1, 2], [3, 4, 5], [6, 7, 8],
        [0, 3, 6], [1, 4, 7], [2, 5, 8],
        [0, 4, 8], [2, 4, 6]
    ]
    for a,b,c in wins:
        if board[a] == board[b] == board[c] and board[a] != ' ':
            return True
    return False

# Board für den KI-Prompt
def board_to_string():
    return "\n" + "\n".join([
        f"{board[0]} | {board[1]} | {board[2]}",
        f"{board[3]} | {board[4]} | {board[5]}",
        f"{board[6]} | {board[7]} | {board[8]}"
    ])

# Anfrage an Llama3.2 schicken für den KI-Zug als O
def get_ai_move():

    prompt = f"""
                Hier ist das Spielbrett (X = Mensch, O = du): {board_to_string()}
                Deine Aufgabe:
                Gib bitte nur eine Zahl von 1 bis 9 zurück. Wähle eine Zahl, die noch nicht belegt ist.
             """

    system_message = {
        "role": "system",
        "content": "Du bist ein Tic Tac Toe-Spieler als O. Wähle die nächste freie Position von 1 bis 9, um zu gewinnen oder zu blockieren."
    }

    user_message = {
        "role": "user",
        "content": prompt
    }

    messages = [system_message, user_message]
    response = ollama.chat(model="llama3.2", messages=messages)
    content = response['message']['content'].strip()

    # Zahl zu extrahieren
    try:
        move = int(''.join(filter(str.isdigit, content)))
        return move
    except:
        return None

turn = 0
game_running = True
has_winner = False

```

```
while game_running:
    os.system('cls' if os.name == 'nt' else 'clear')
    display_board()

    current_player = 'X' if turn % 2 == 0 else 'O'

    if current_player == 'X':
        try:
            move = int(input("Dein Zug 1-9: "))
            if move < 1 or move > 9 or board[move - 1] != ' ':
                print("Ungültiger Zug. Bitte erneut.")
                input()
                continue
        except ValueError:
            print("Bitte gib eine Zahl ein.")
            input()
            continue
    else:
        print("KI denkt...")
        move = get_ai_move()
        while move is None or move < 1 or move > 9 or board[move - 1] != ' ':
                print(f"Ungültiger KI-Zug {move}. Versuche erneut...")
                move = get_ai_move()
        print(f"KI wählt Feld: {move}")

    board[move - 1] = current_player
    turn += 1

    if check_winner():
        has_winner = True
        game_running = False
    elif turn == 9: game_running = False

# Finale Anzeige
os.system('cls' if os.name == 'nt' else 'clear')
display_board()

if has_winner:
    winner = 'O' if turn % 2 == 0 else 'X'
    print(f"Spieler {winner} gewinnt!")
else:
    print("Unentschieden.")

print("Danke fürs Spielen!")
```

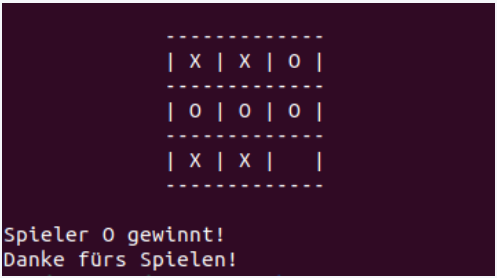
Ergebnisse

Ergebnisse der Spielrunden

Spieler X gewinnt

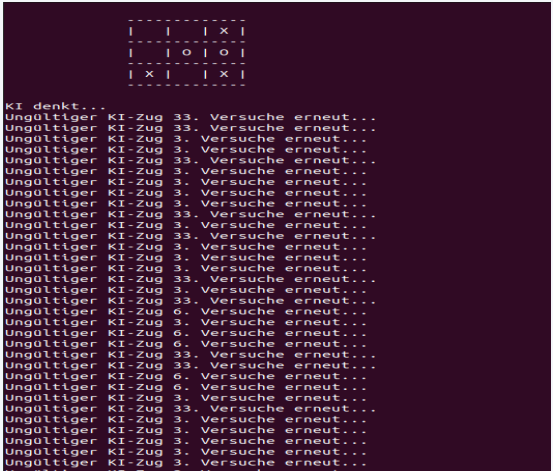


Spieler O gewinnt



Verzögerungen bei der KI-Zugauswahl

In seltenen Fällen benötigte Llama3.2 bis 30 Sekunden für eine gültige Antwort. Typischerweise lag die Reaktionszeit bei 2-4 Sekunden pro Zug.



Zusammenfassung

Schlussfolgerung

- Llama3.2 ist primär ein Sprachmodell ohne Spielstrategie.
- API-Aufrufe führten zu verzögerten Antwortzeiten.
- Es gab keine Implementierung für einen vorzeitigen Spielabbruch.

Referenzen

Jan-Torsten Milde (2025)
schueler_tag_KI
https://github.com/drmilde/schueler_tag_KI

OpenAI (2025)
ChatGPT: A Large Language Model by OpenAI
https://openai.com/chatgpt

YouTube
Verschiedene Tutorials zu Python und Tic-Tac-Toe
https://www.youtube.com

Fragen ? Stellen : Danke

Tumar Chegebaeva

Multimodale Mensch Computer Interaktion

July 14, 2025