GOR 2025 Workshop: Structured Information Extraction with LLMs

Paul Simmering

Q Agentur für Forschung GmbH

Table of Contents

1 Introduction

1.1 Q Agentur für Forschung

- Market research agency from Mannheim
- Qualitative and qualitative research
- Using language models since 2019 for social media and review analysis
- I’m a senior data scientist at Q

This workshop builds on the GOR24 presentation “Where do LLMs fit in NLP pipelines?” by my colleague Dr. Paavo Huoviala and me.

1.2 Goals for today

- Get hands-on experience with structured output tasks
- Get an overview of model and tool choices
- Share experiences and use cases

1.3 Pre-requisites

- Basic Python knowledge (easy to pick up for R users)
- Need API access to a large language model provider: OpenAI, AzureOpenAI Service, Google Gemini, Anthropic, Cohere, or a litellm-compatible backend
- Ability to run Python code on your machine or Google Colab
- Ideally do the set-up before the workshop

1.4 Setup for today

- Slides and example code is available on GitHub
- Please ask questions at any time
- 4 exercises, use my example cases or bring your own
- 2 discussions

1.5 Installation

Clone the repository:

```
git clone qagentur/gor-2025-workshop
```

Create a venv and install the required packages:

```
python -m venv venv
source venv/bin/activate
pip install instructor
```

Set the API key for your language model provider:

```
export PROVIDER_API_KEY=your-api-key
```

2 LLMs and structured output

2.1 Structured output tasks

- Text classification
- Named entity recognition
- Sentiment analysis
- Relation extraction
- Summarization
- Question answering

2.2 Why use an LLM for structured output?

- Just a few examples are needed
- Do any task with the same model
- Do multiple tasks at once
- Analyze text, images and audio in the same way
- Easy to use via an API

2.3 Model choices

- Size/Cost vs performance
- Open and closed source choices

2.4 JSON mode

2.5 Function calling

2.6 Constrained generation

2.7 instructor and Pydantic

Instructor is a Python package for structured output tasks with language models.

2.8 Exercise 1: Extracting structured information

Pick a structured output task and try to solve it with your language model provider

- Use a Pydantic model to describe the desired output
- Find a case where the model fails to give the right answer

3 Learning from examples

3.1 Few-shot learning

3.2 Exercise 2: Few-shot learning

Add examples to your structured output task

3.3 Fine-tuning

3.4 Discussion: Fine-tuning vs few-shot learning

- When would you use fine-tuning over few-shot learning?

4 Evaluation

4.1 Metrics

4.2 Exercise 3: Evaluation

Evaluate your model on a test set.

- compare different models
- compare different few-shot examples

4.3 Discussion: Classic NLP vs LLMs

- When would you use classic NLP methods over LLMs?

5 Efficiency

5.1 Prompt caching

5.2 Caching

5.3 Cost control

5.4 Parallelization

5.5 Exercise 4: Efficiency

Optimize your pipeline for cost and speed.

- check the cost of your API calls
- parallelize your API calls
- cache your prompts

5.6 Contact

Website: teamq.de

Github: github.com/qagentur

Contact me: paul.simmering@teamq.de

We offer market research services and consulting on AI projects.