Advanced NLP - The Road to LLMs

Prof. Dr. Richard Sieg
TH Köln IWS - WS 25/26

Goal of this Course

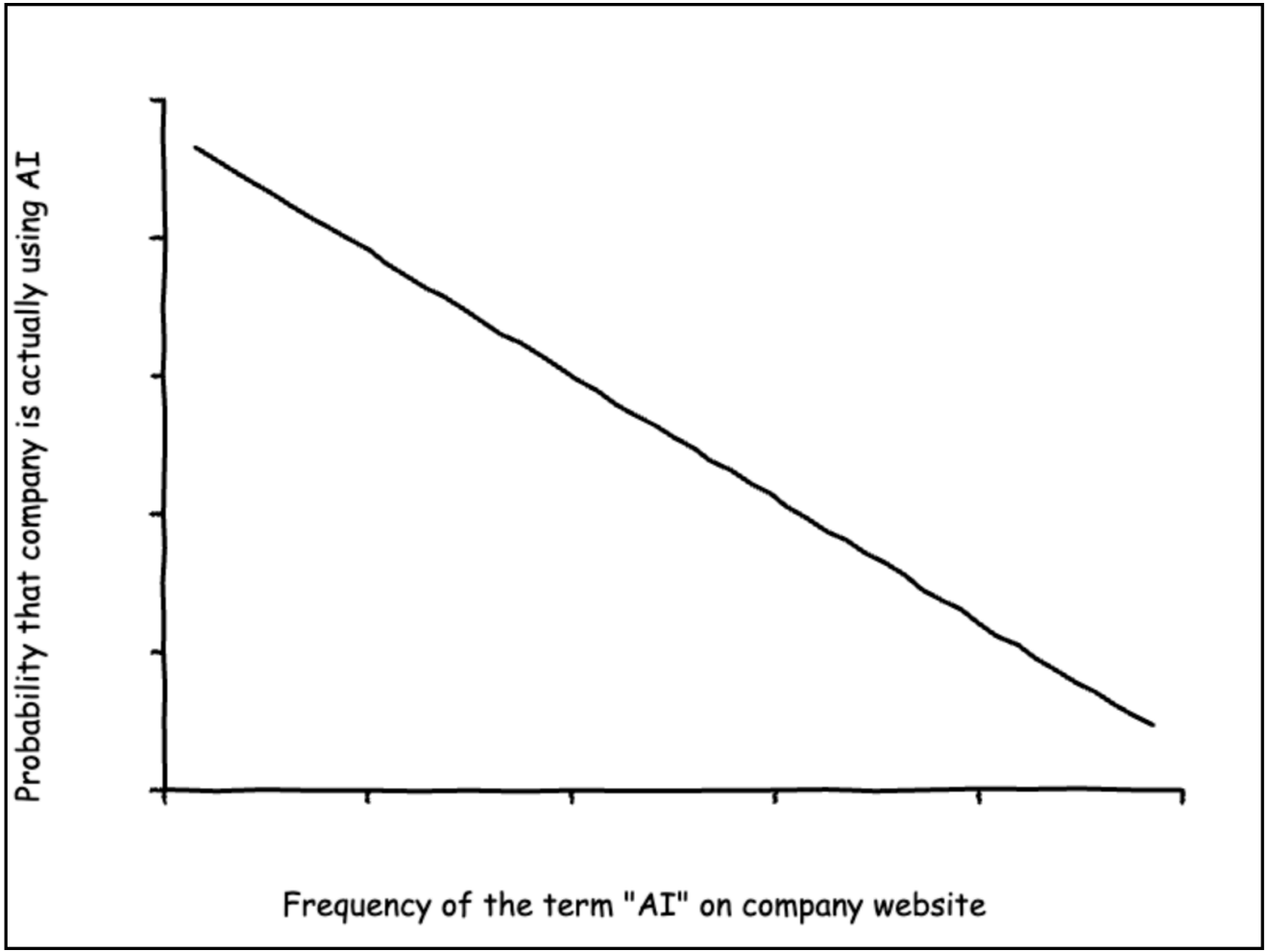
Insights into the development of the first LLMs and their basic architecture & design.

How to use and steer LLMs programmatically for research & industry.

Sensitivity for shortcomings of LLMs: Bias, hallucinations...


With great power
comes great responsibility.

Ravin Kumar - Models as Tools


Probability that company is actually using AI
Frequency of the term "AI" on company website

Agenda

01. Organization

02. NLP Recap

03. Text Simplification

04. Set-Up & Tutorial

05. Appendix

01. Organization

About Me

- First term as full-time Professor for NLP & LLMs
- Since 2023 external lecturer for NLP @ IWS
- Mathematics Background
- NLP & ML Engineer for eight years
- Many years of university level teaching
- richardsieg.github.io

Shall we do one session via Zoom?

In January I would like to have at least two extended sessions
(One being the presentations Jan 15th)

Schedule

Lecture	Date	Topic
1	03.12.2025	Introduction & NLP Recap
2	04.12.2025	RNNs and LSTMs
3	10.12.2025	Attentions & Transformers
4	11.12.2025	Transformer Based Models
5	17.12.2025	Hackathon / Check-In
6	18.12.2025	LLM Architecture
7	07.01.2026	LLM Engineering
8	08.01.2026	Hackathon / Check-In
9	14.01.2026	LLM Shortcomings
10	15.01.2026	Final Presentations

Requirements & Grading

The exam and grade consists of

- A short (max 2 pages) report (pdf format) on the implementation, results and the work done by the team members, along with well-documented notebooks and code (50%, per team)
- A 10 minute oral presentation per team member on their individual work on the 15th of January 2026 (50%, individual)

The final code and report has to be ready in your team's GitHub Repo by the 16th of January 2026, 6pm.

Presentation

- Oral Presentation of the code and evaluation results per individual team member
- No slides, just code
- Approx 10 minutes per presentation
- Split up your team in a meaningful way
- Small Q&A after each presentation

Find teams and fill out the form

Teams

- Team 1
- Team 2
- Team 3
- Team 4

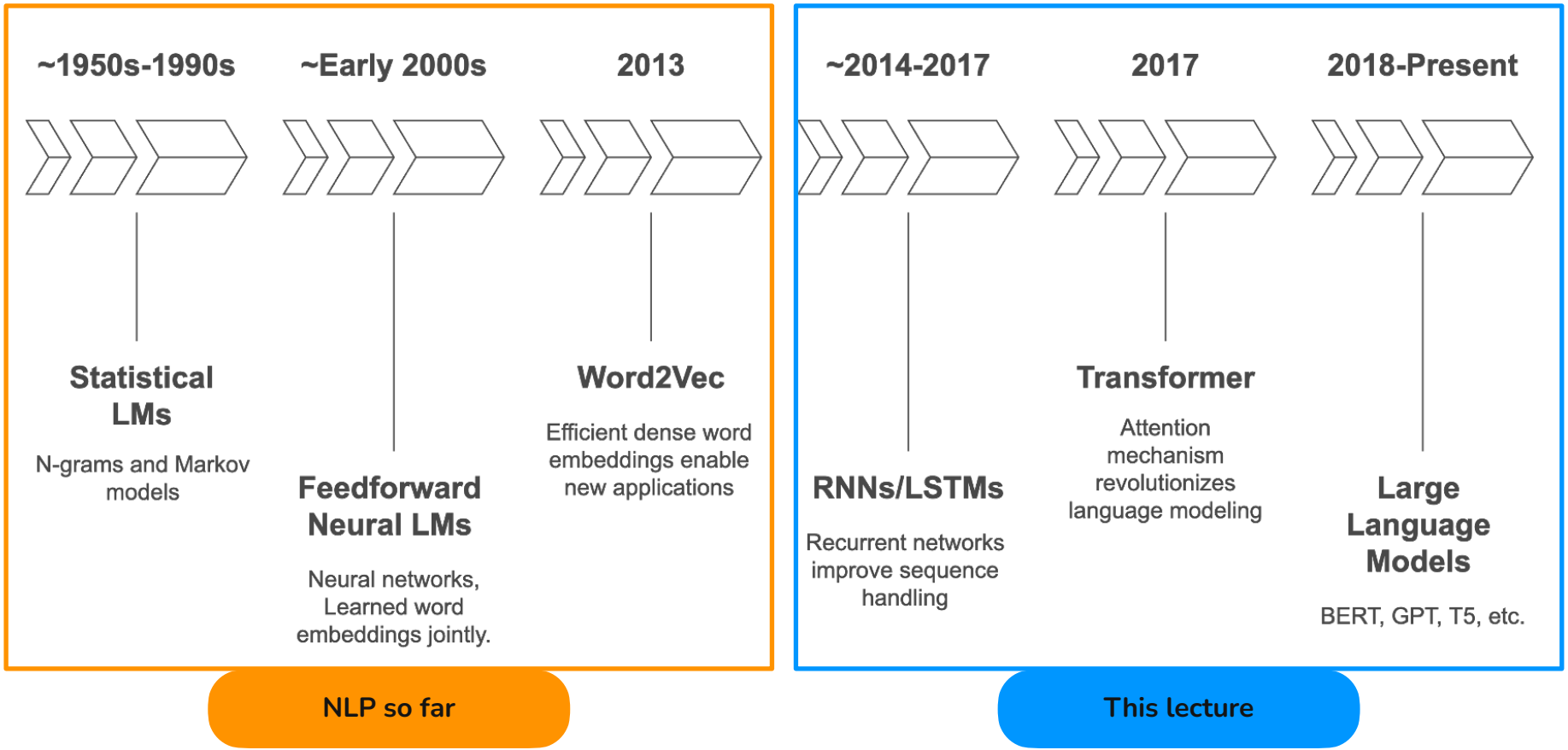
- Team 5
- Team 6
- Team 7
- Team 8

02. NLP Recap

NLP so far

- Text Processing (spaCy, RegEx)
- Relations and semantics
- Frequency Based Embeddings (BoW, tf-idf)
- Prediction Based Word Embeddings (word2vec)
- Information extraction
- Text Classification
- Language Modeling

History of Language Models



Back to the Future

Rules

Text Processing (NLTK, SpaCy, RegEx..)

Machine/Deep Learning

Supervised Learning by examples

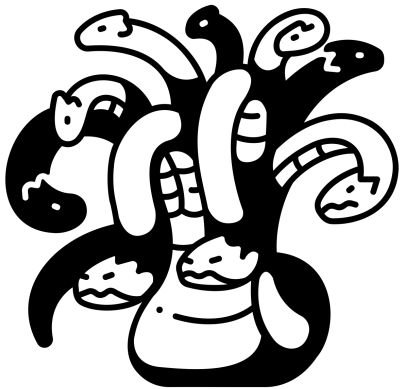
Instructions for LLMs

Rules

The many-headed AI Hydra

One particular system or model

Large Language Models



"AI" means…

... and gets confused with Machine Learning

The research field AI

Artificial General Intelligence

You shall know a word by
the company it keeps.

J.R. Firth 1957

Frequency Based Embeddings

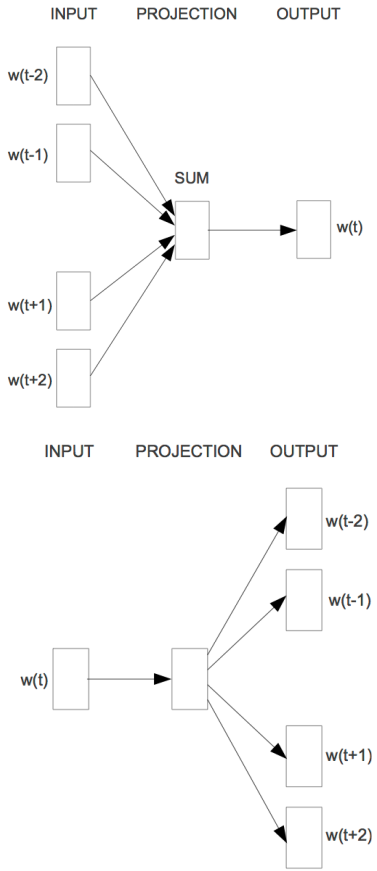
- Given: A number of *documents*
- **Bag of Words**: Count how many times a word appears in a document
  - Vector representation of one document
  - Vector dimension: size of vocabulary (unique words)
- **Term Frequency - Inverse Document Frequency**

$$\mathrm{tf}(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \qquad \mathrm{idf}(t,D) = \log \frac{|D|}{|\{d \in D : t \in d\}|}$$

$$\mathrm{tfidf}(t,d,D) = \mathrm{tf}(t,d) \cdot \mathrm{idf}(t,D)$$

Predictive Word Embeddings

- Learn a vector representation of a word by predicting…
  - the word itself based on its context (Continuous Bag of Words)
  - the context words based on a target word (Skip-Gram)
- Word2vec, fastText, GloVe
- Issue: generates fixed embeddings, neural network itself cannot be used for further task specific fine-tuning



Discriminative and Generating NLP Tasks

Discriminative Tasks

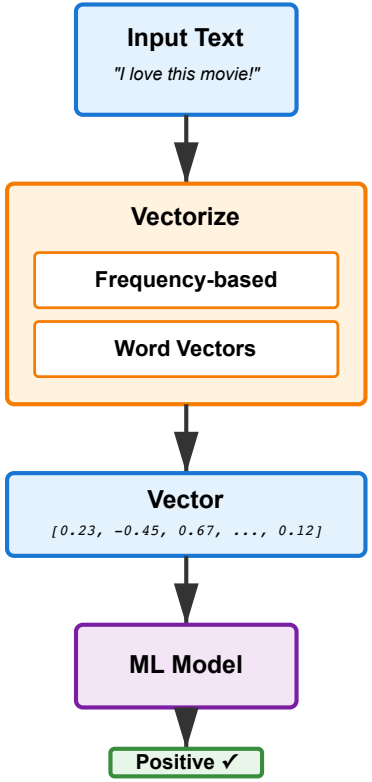
- Text Classification (e.g. sentiment)
- Named Entity Recognition
- Part of Speech Tagging
- Semantic Role Labeling

Generative Tasks

- Chatbots
- Text Completion
- Machine Translation
- Question Answering
- Summarization
- **Simplification**

Text Classification (so far)

- Given an input text vectorize it by
  - using frequency based embeddings
  - or combining the individual word vectors (e.g. average)
- Take this vector as an input for a classical ML model and train
  - Logistic regression, SVM, Naive Bayes, NN ...



Language Modeling

- A *language model* is a statistical model that assigns a probability to a sequence of words (more precisely *tokens*)

  $p(w_1, \ldots, w_m)$

- Related task: Compute the probability of the next token given a sequence of previous tokens

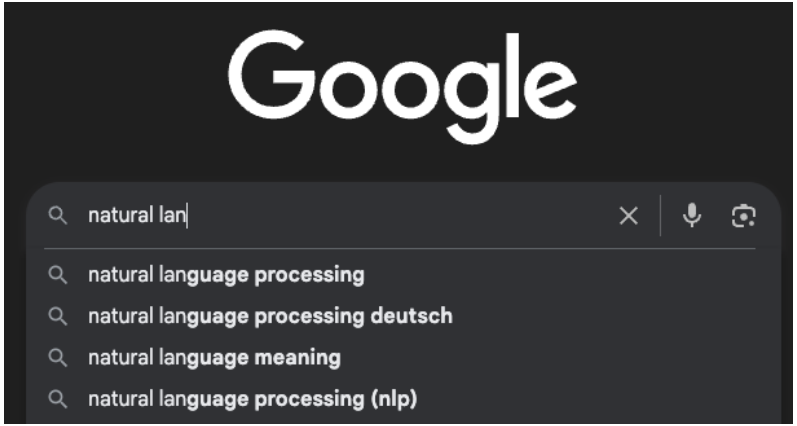
  $p(w_t \mid w_{t-1}, \ldots, w_{t-n-1})$

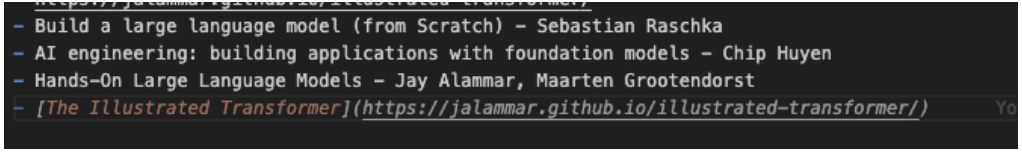
- *Chain Rule*

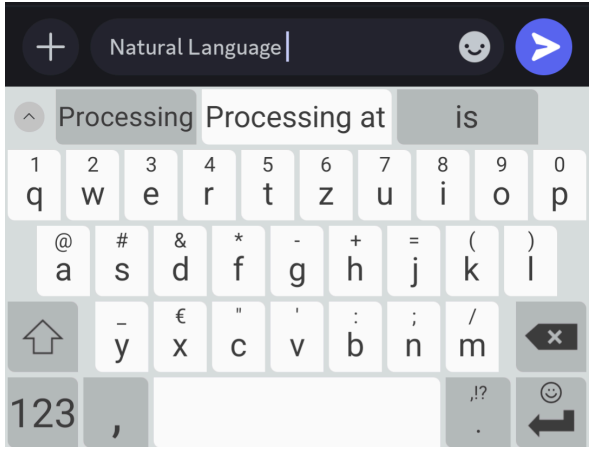
  $$p(w_1, \ldots, w_m) = \prod_{i=1}^{m} p(w_i \mid w_1, \ldots, w_{i-1})$$

- p(be water my friend) = p(be) * p(water | be) * p(my | be water) * p(friend | be water my)

Language Modeling is everywhere







Markov Assumption

- We can approximate the conditional probability by only look at the past n tokens (n-gram model)
  - p(friend | be water my) ≈ p(friend | my) (*bigram*)
- Bigram model: Predict the next word by only looking at the previous word

$$p(w_i \mid w_1, \ldots, w_{i-1}) \approx p(w_i \mid w_{i-1})$$

- Calculation is based on counting frequencies

$$p(w_t \mid w_{t-1}, \ldots, w_{t-n-1}) = \frac{count(w_{t-n+1}, \ldots, w_{t-1}, w_t)}{count(w_{t-n+1}, \ldots, w_{t-1})}$$

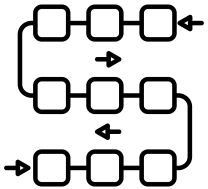
n-gram Language Modeling

- Markov Assumption can be used to build a Language Model by computing n-gram probabilities over a large corpus

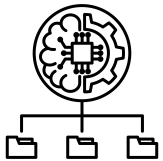
$$p(w_t \mid w_{t-1}, \ldots, w_{t-n-1}) = \frac{count(w_{t-n+1}, \ldots, w_{t-1}, w_t)}{count(w_{t-n+1}, \ldots, w_{t-1})}$$

- **Issue 1: Sparsity -** If a sequence of words rarely or never occurs in the corpus, the probability is almost 0
- **Issue 2: Storage -** We need to store all occurrences of all sequences of tokens
- Increasing the window n makes these issues worse. More than 5 impractical

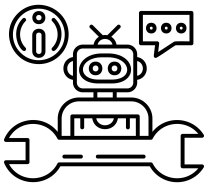
The 3 Ingredients of LLMs



Process long sequences and context



Efficient training on huge datasets



Follow (human) instructions

03. Text Simplification

Text Simplification

- The task of simplifying text while preserving meaning.
- Can be general or context specific, e.g. “fourth graders” or “Leichte Sprache” or “A2 reading level”.
- Before LLMs
  - Fine-Tuned Sequence to Sequence models (see next lecture)
  - Word tagging and replacement
- Out-of-the-box LLMs already match or are superior to these approaches.

➡ Major parts of research is focused on utilizing LLMs nowadays.

Measuring Simplicity

- FKGL: US Navy based score that measures relation of syllables, words, and sentences
- SARI: Needs source and reference sentence and measures the suitability of words that were added, deleted, and kept
- BLEU: Score from Machine Translation; measures syllable overlap with reference sentence
- BERTScore: Score from Machine Translation; compares the BERT sentence embeddings of simplification and reference
- SLE: RoBERTa model fine-tuned to score simplicity

Source

Owls are the order Strigiformes, comprising 200 bird of prey species.

Reference Simplification

An owl is a bird. There are about 200 kinds of owls.

Predicted Simplification

Owls are a group of birds with about 200 species that hunt for food.

The challenge

Fourth Workshop on Text Simplification, Accessibility and Readability

TSAR 2025 @ EMNLP 2025

Organizers

Matthew Shardlow

Senior Lecturer at Manchester Metropolitan University, UK

Fernando Alva-Manchego

Lecturer at Cardiff University, UK

Kai North

Senior Data Scientist at Cambium Assessment, USA

Regina Stodden

Postdoctoral Researcher at University of Bielefeld in the LLM4KMU Project, Germany

Horacio Saggion

Chair in Computer Science and Artificial Intelligence and Head of the LaSTUS Lab in the TALN-DTIC, Universitat Pompeu Fabra

Nouran Khallaf

Postdoctoral Research Fellow at University of Leeds, UK

Akio Hayakawa

PhD Student at Universitat Pompeu Fabra, Spain

The challenge

- All links, data, and additional information in our repo
- Trial data already available, test data (without simplification) by end of the term
- Simplify a given sentence for a specific CEFR reading level (A2 or B1)
- **Important:** Submit and discuss at least one approach using seq2seq models (LSTMs, T5, BART) and one approach using LLMs
- Data contains source statement, simplified version, and reading level
- Three evaluation metrics (script is also in repo) - all BERT based
  - CEFR classifier for predicting correct CEFR level
  - Meaning preservation (using BERTMean model)
  - Semantic similarity (using BERTScore)

Text Simplification Datasets for Training

- https://github.com/jantrienes/text-simplification-datasets
- ASSET (Fernando Alva-Manchego, Louis Martin)
  - We use this one in the next lecture

04. Set - Up

uv

- New Package Manager that has replaced all others (pip, poetry etc) and is very fast
- Management of packages in a `pyproject.toml`
- Python Installation: `uv python install 3.12`
- First Installation and venv creation: `uv sync`
- Run scripts: `uv run ...`
- Add package: `uv add ...`

marimo

- New alternative to Jupyter Notebooks
- Lives in Python files (-> better Diffs & Imports)
- Reactive: Dependent cells are automatically executed
- Interactive: Many possible Input Options
- Coding Assistant via GitHub Copilot & LLM APIs (OpenAI etc)
- Editor: In Browser or VSCode/Cursor Extension
- Start editor: `(uvx) marimo edit`
- Run as Web App: `(uvx) marimo run notebook.py`

GitHub

- One main GitHub repository for all slides and notebooks
- Please do a fork, invite your team members and me, and rename it so it contains your team number
- When you are all set, run `uv sync` and `uvx marimo edit`
- Remember to register for GitHub Education (Copilot Pro for free)

GPUs

- For this course we have two options:
- marimo's platform molab gives us free GPU 🎉 T4, A100, H100
- The institute has access to clusters in Mainz (A100 1-8GB)
  - Mostly JupyterLab, theoretically it's possible to start marimo
  - marimo notebooks can be converted to Jupyter ones
- I will invite you to both of them.
  - For molab, I need your email address which you use for GitHub

05. Appendix

STUDENT ELECTOIN

rhAInland Community - Join our Discord!

Join us @ Cologne IR

Hands-on experience in applied research

- Deepen skills you learned during study
- Programming (Python), data annotations, data cleaning, literature research, etc.
- Find spin-off topics for your thesis

Hard facts

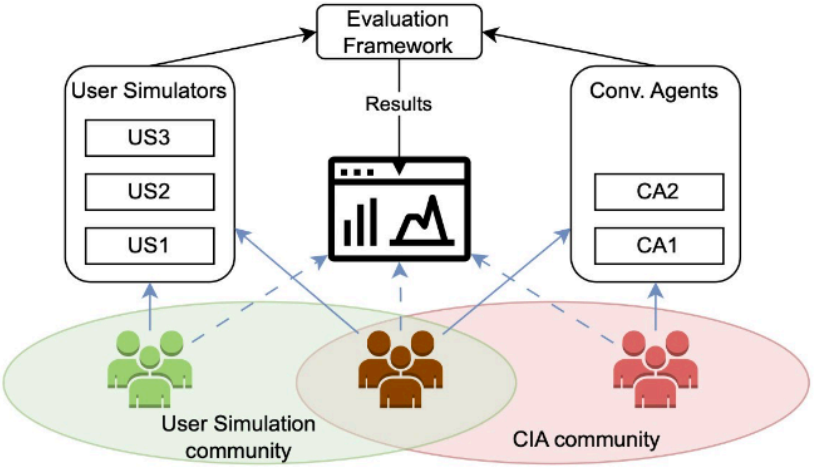
- Flexible/student-friendly working hours
- 8-17 hours/week and €15 hourly
- Up to 12 months contract, extension possible

Coffee and Kamelle

- Nice & friendly place to work + great team!
- apply@cir-group.org

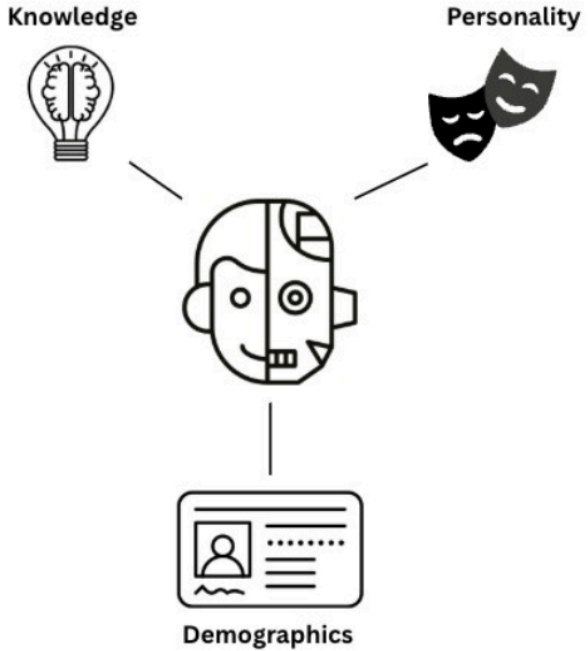
Project 1: Creation of a shared task using SimLab

- The objective of this project is to create a **shared task** using the **SimLab platform** (https://github.com/iai-group/simlab).
- The shared task will benchmark **user simulators** and **conversational information access (CIA) agents**.
- You will gain experience in **research design**, **community engagement**, and technical aspects of **implementing a shared task**.

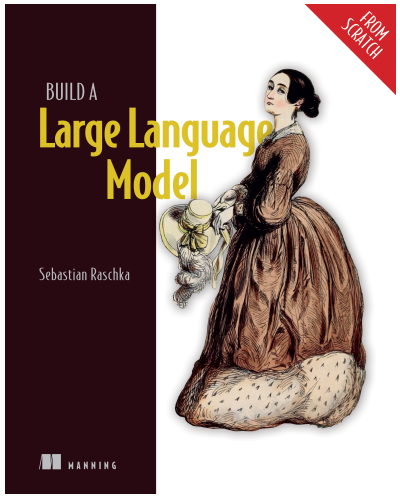


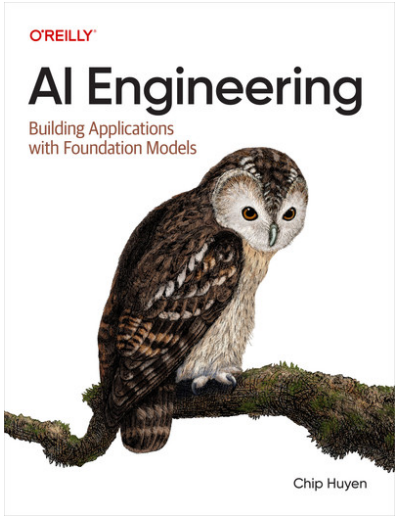
Project 2: Expansion of UserSimCRS toolkit

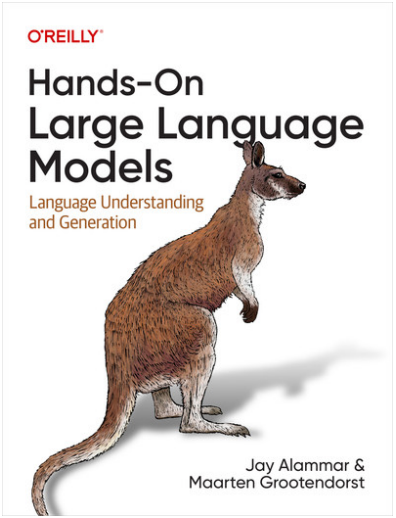
- The objective of this project is to contribute to UserSimCRS v3 (https://github.com/iai-group/UserSimCRS)
- Focus on enhancing and expanding features to **model user's knowledge, preferences, and personality** → More realistic simulation
- You will gain experience in **open-source software development**, **evaluating conversational recommender systems** and **user modelling**.



Some Literature







Folks to follow

- Godfathers of AI: Geoffrey Hinton, Yoshua Bengio, Yann LeCun
- Andrew Ng - Coursera founder, influential AI figure, has newsletter (The Batch)
- Andrej Karpathy - LLM guru, great YT videos, former OpenAI, now Tesla
- Chip Huyen
- Jay Alammar
- Sebastian Raschka
- Vincent Warmerdam - YouTuber, great talks, former spaCy, now marimo
- Ines Montani - Founder of spaCy
- *For more newsletters and blogs see richardsieg.github.io*