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.

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 meet 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	Encoder & Decoder 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

- Code implementation of all discussed milestones (30%, per team)
  - Code needs to be functioning and explained via comments
- A short (max 2 pages) report (pdf format) on the implementation, results and the work done by the team members (20%, 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

Back to the Future

Rules

Text Processing
(NLTK, SpaCy, RegEx..)

Machine/Deep Learning

Supervised Learning by examples

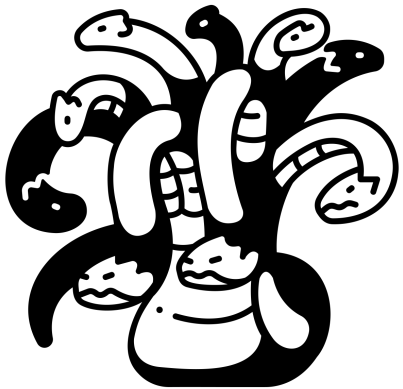
Rules

Instructions for LLMs

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

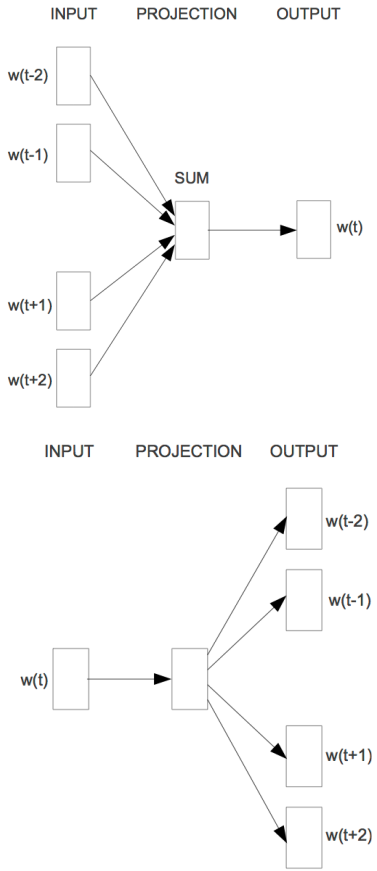
- Given: A number of *documents*
- **Bag of Words**: Count how many times a word appears in a document
  - Vector representation of one document
  - Vector dimension: size of vocabulary (unique words)
- **Term Frequency - Inverse Document Frequency**

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

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

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

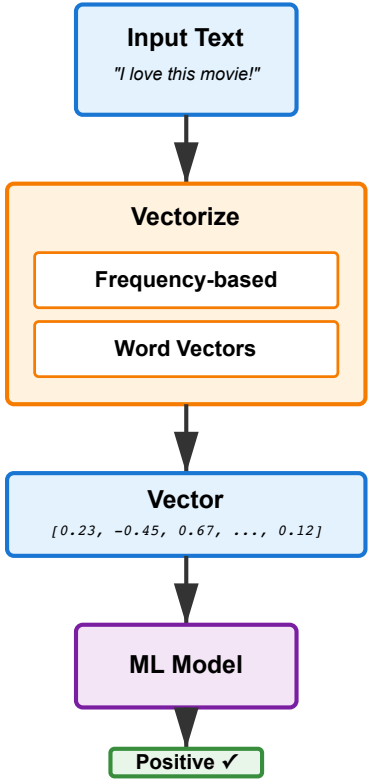
- Text Classification (e.g. sentiment)
- Named Entity Recognition
- Part of Speech Tagging
- Semantic Role Labeling

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)

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

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

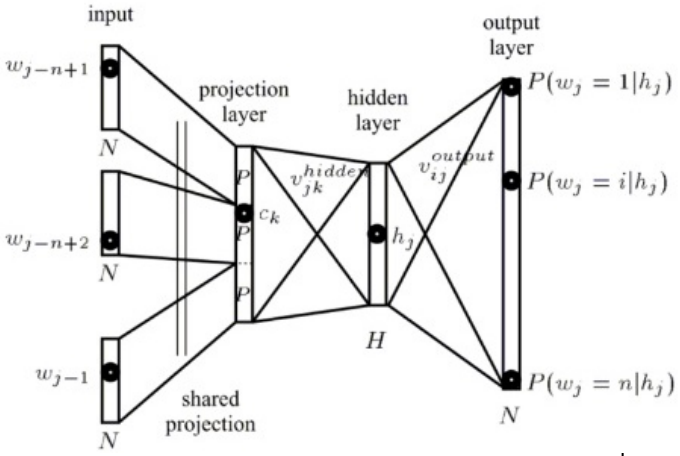
- Calculation is based on counting frequencies

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

Language Modeling with Neural Networks

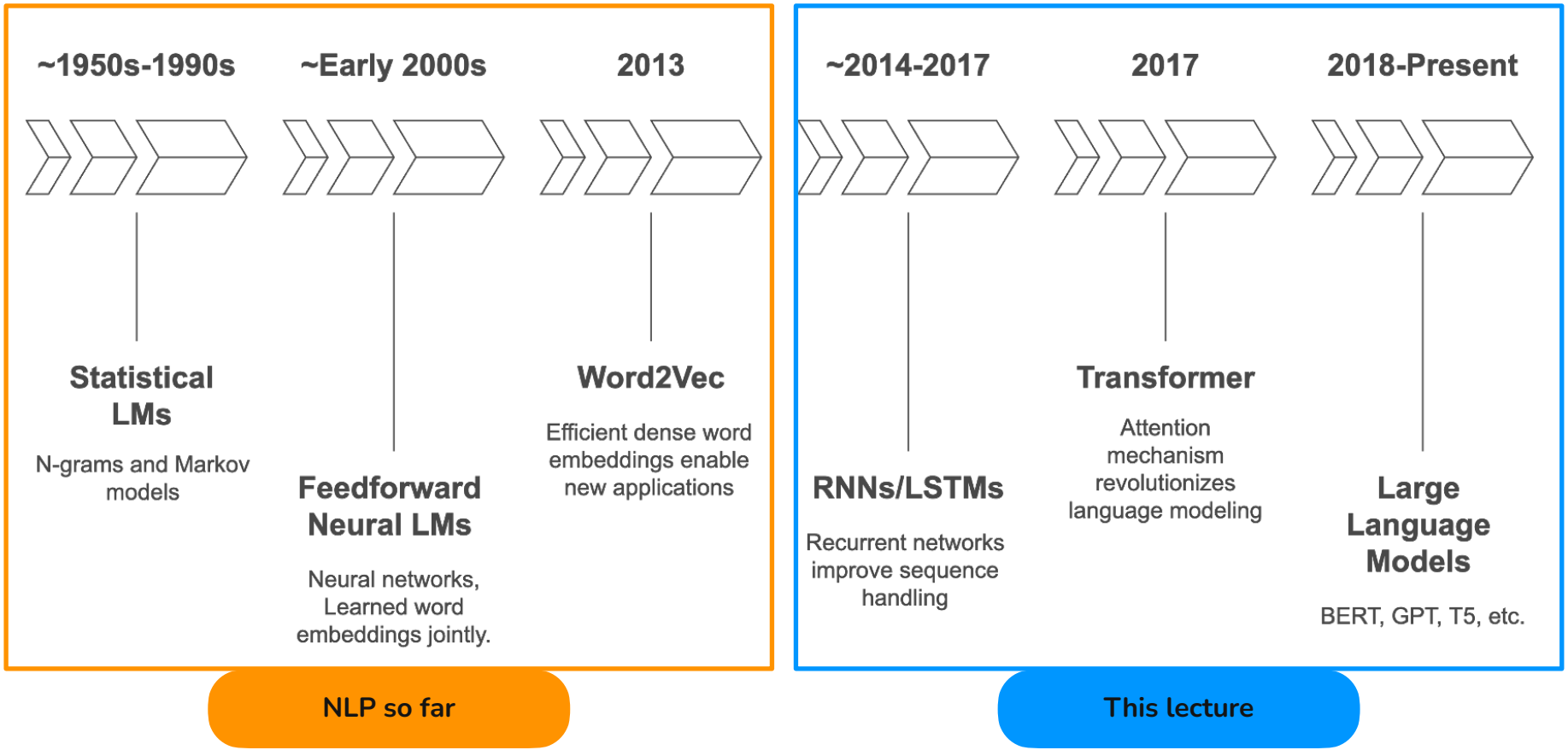
- The conditional probability can be expressed as a *softmax* layer inside a neural network

$$p(w_t \mid w_{t-1}, \ldots, w_{t-n-1}) = \frac{\exp(h^T v'_{w_t})}{\sum_{w_i} \exp(h^T v'_{w_i})}$$



- h: *context vector* (output of the hidden layer on $w_{t-1}, \ldots, w_{t-n-1}$)
- v'_{w_t}: the embedding to be learned for the token/word w_t
- Objective function: Maximize probability for the correct word

History of Language Models



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 “A1 reading level”.
- Before LLMs
  - Fine-Tuned Sequence to Sequence models
  - 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

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: `marimo edit`
- Run as Web App: `marimo run notebook.py`

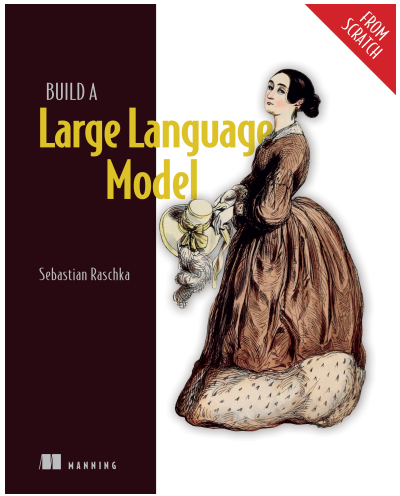
GitHub

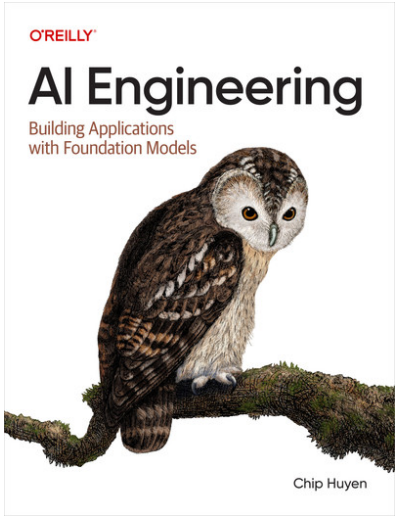
- One main GitHub repository for all slides and notebooks
- Please do a fork, invite your team members and me, and rename it so it contains your team number
- When you are all set, run `uv sync` and `uv run marimo edit`
- Remember to register for GitHub Education (Copilot Pro for free)

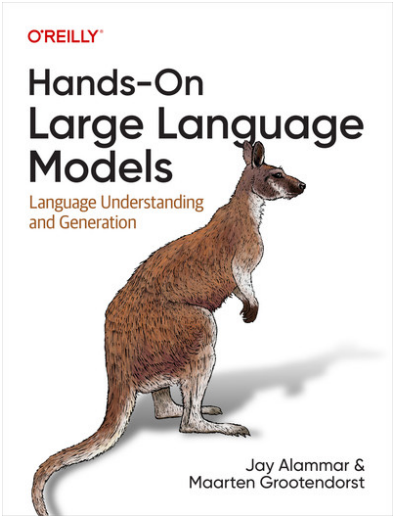
05. Appendix

rhAInland Community - Join our Discord!

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*