

Dialogue & Text Generation with Transformer Models

Ondřej Dušek

JSALT Workshop

13. 6. 2025



Charles University
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics



unless otherwise stated

Dialogue Systems

- Definition: A (*spoken*) dialogue system is a **computer system designed to interact** with users **in** (*spoken*) **natural language**
 - Wide – covers lots of different cases

Task-oriented

- focused on completing a certain task/tasks
 - booking restaurants/flights, finding bus schedules, smart home...
- most actual DS in the wild & our focus today
- (typically) **single/multi domain** – 1/few topics

Non-task-oriented

- chitchat – social conversation, entertainment, ...
 - persona, gaming the Turing test
- typically **open-domain** – talk about anything

Real-life dialogue systems: LLM Chat

- ChatGPT(3.5/4/4o/o1/o3...), Claude..., Llama, Mistral, Qwen
 - closed models (API-only) vs. open (you can download them)
- Ask anything, get some reply
 - impressive skills in text editing, coding, ...
 - reply quality varies (hallucination)



Who is Ondřej Dušek?

14, depending how you count



Ondřej Dušek is a researcher and academic in the field of natural language processing (NLP). He is currently an Assistant Professor at the Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic.

Dušek's research focuses on developing computational models and algorithms for natural language generation, dialogue systems, and machine learning approaches for NLP tasks. He has published several research papers in top-tier NLP conferences and journals, and he has also served as a reviewer and program committee member for these conferences.

In addition to his research, Dušek is also actively involved in teaching and mentoring students in the field of NLP. He has supervised several Master's and PhD theses, and he teaches courses on topics such as computational linguistics, NLP, and machine learning.

no other tasks than generation & dialogue

no PhD graduate at the time

I only teach dialogue systems



Compose a limerick on how to exit Vim.



There once was a programmer in Vim
Whose exit strategy was looking grim
They tried to escape
But couldn't find the right shape
Until they remembered to type ":q!" and win!



Nice, now try it while speaking like a pirate



Arrr, listen ye mateys to me tale,
Of a sailor who be stuck in Vim's jail,
He couldn't leave the text editor,
No matter how he did endeavour,
But then he remembered ":q!" and set sail!

Real-life dialogue systems: Virtual Assistants

- Google, Amazon, Apple..., Rhasspy (open-source)
- Hardware devices, but mostly work online
 - good microphones + low-spec computer
- Huge knowledge bases
 - Google: combined with web search
- Lots of domains programmed in, but all by hand
 - integration with a lot of services (calendar, music, shopping, weather, news...)
 - you can add your own (with limitations)
- Can keep some context
- Conversational capabilities limited



Amazon Echo



Google Nest



Apple HomePod

Task-oriented Dialogue Example

(Budzianowski et al., 2018)

<https://aclanthology.org/D18-1547/>

- MultiWOZ: benchmark for task-oriented dialogue in multiple domains
 - hotels, restaurants, attractions, trains, taxi, police, hospital
 - domains are connected (e.g. taxi from hotel to attraction)
 - 10k dialogues, extensive annotation (but noisy!)

- user:** I am looking for a train from Cambridge to London Kings Cross.
state: *{train {departure = cambridge, destination = london kings cross}}*
DB: *{train (70) {...}}*

	[count]	[departure]	[destination]
system:	There are 70	trains departing from Cambridge to London Kings Cross.	What day would you like to travel?
- user:** I would like to leave on Saturday after 18:45.
state: *{train {day = saturday, departure = cambridge, destination = london kings cross, leave at = 18:45}}*
DB: *{train (3) {arrive by = 19:51,21:51,23:51; id = TR0427,TR0925,TR4898; leave at = 19:00,21:00,23:00; ... }}*

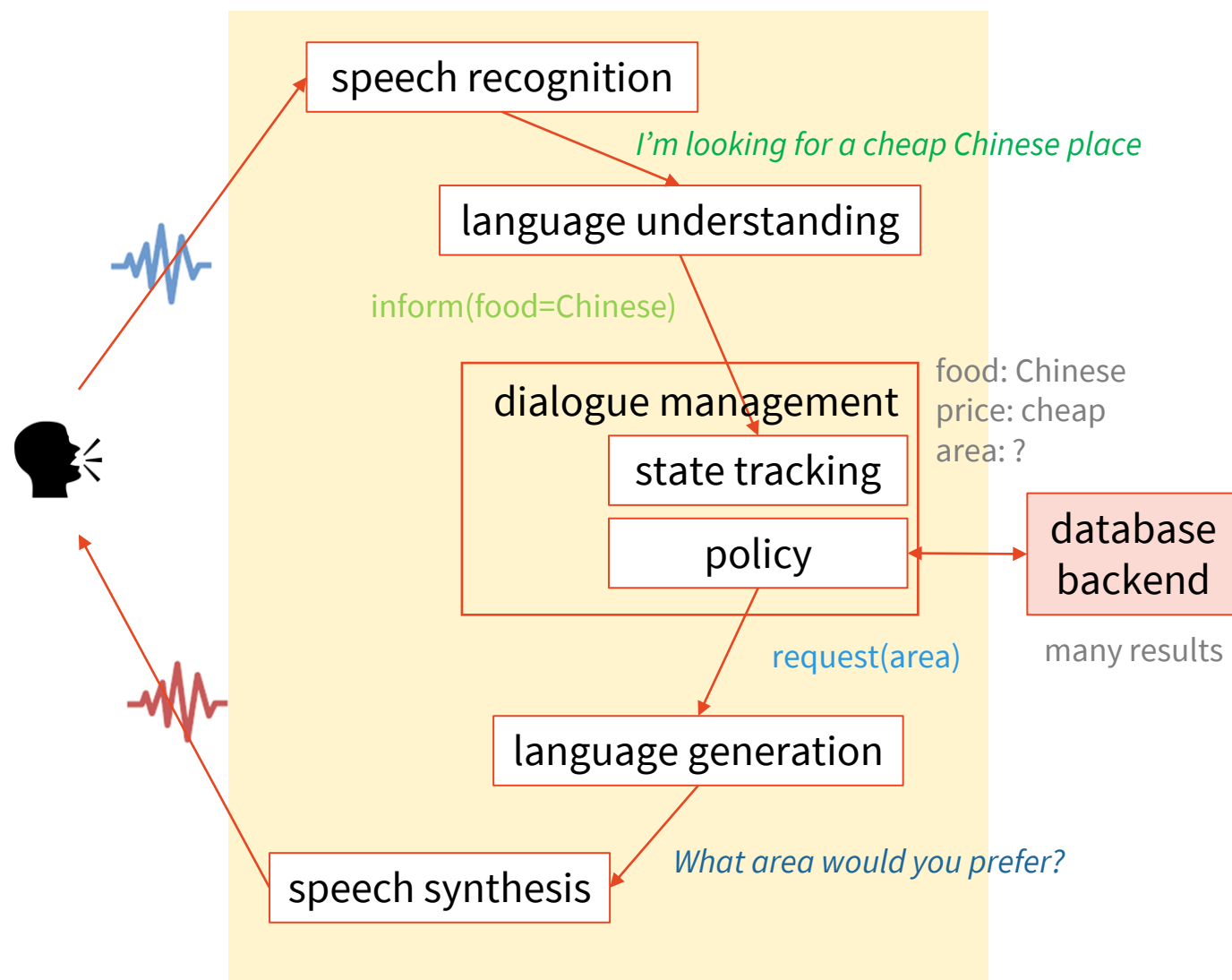
	[id]	[leave_at]	[arrive_by]
system:	TR0427	leaves at 19:00	and arrives by 19:51 . Would you like to book this train?
- user:** Yes, I would like to book it for eight people.

	[reference]
system:	I have booked it for you. Your reference number is 00000057. Is there anything else I can help you with?
- user:** I am also looking for an expensive restaurant in the centre.
belief: *{restaurant {area = centre, price range = expensive} train {...}}*
DB: *{restaurant (33) {area = centre (33); name=Curry Garden, ...; ...} ...}*

	[count]	[price_range]	[area]
system:	There are 33	expensive restaurants in the centre.	Is there a particular type of food you would like?

Dialogue Systems Architecture

- traditional DS pipeline:
 - ASR: voice → text
 - **NLU: text → meaning**
 - **DM: meaning → reaction**
 - **NLG: reaction → text**
 - TTS: text → voice
- backend
 - needed for anything better than basic chit-chat
- text-based systems:
NLU→DM→NLG only



Natural/Spoken Language understanding (NLU/SLU)

- **Extracting the meaning** from user utterance text

Chinese city centre

I've been wondering if you could find me a restaurant that has Chinese food close to the city centre please

- Converting into a structured semantic representation

- **dialogue acts:**

- **intent**/act type (*inform, request, confirm*)

inform(food=Chinese, price=cheap)

- **slot**/entity/attribute (*price, time...*)

request(address)

- **value** (*11:34, cheap, city center...*)

- = you need **intent detection + slot-value tagging**

- other, more complex representations – e.g. syntax trees, predicate logic

- You can get far with keywords/regexes (for a limited domain)
- **Intent classification**
 - Transformers, PLMs: typically over 1st input element (start-of-sentence token)
- **Slot value detection**
 - classification (binary: “is slot value *X* present?”) *I need a flight from Boston to New York tomorrow*
 - **slot tagging** – classify every token *OO OO O B-dept O B-arr I-arr B-date*
 - **BIO/IOB** scheme: beginning (+slot) – inside (+slot) – outside
- **Delexicalization**: replacing slot values by placeholders
 - essentially named entity recognition
 - essentially tagging, but typically done by dictionaries

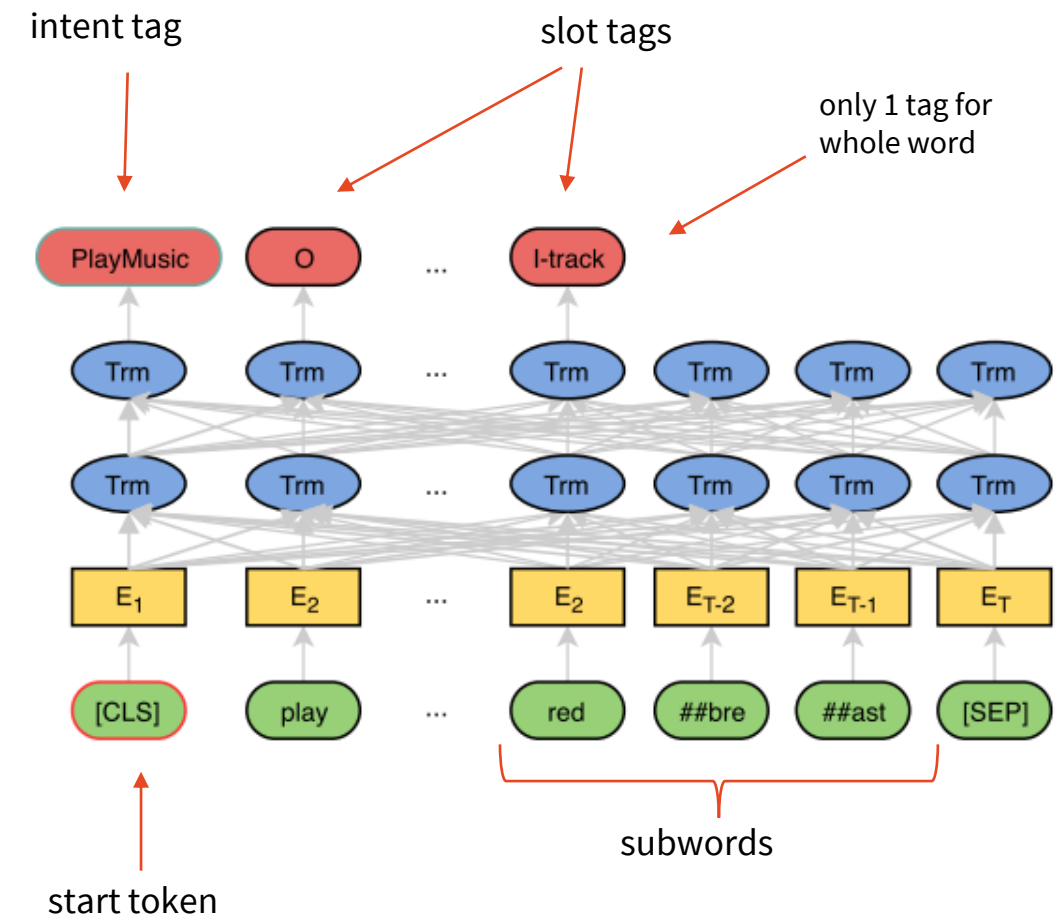
I'm looking for a Japanese restaurant in Notting Hill.
I'm looking for a <food> restaurant in <area>.

I need to leave after 12:00.
I need to leave after <time>.
(= not necessarily 1:1 with slots)

BERT-based NLU

- combined intent-slot
- slot tagging on top of pretrained BERT
 - standard **IOB approach**
 - feed last BERT layers to **softmax over tags**
 - classify only at 1st subword in case of split words (don't want tag changes mid-word)
- special start token tagged with intent
 - again, softmax on top of last BERT layer
- finetune both tasks at once
 - essentially same task, just having different labels on the 1st token 😊
- extension: TOD-BERT
 - pre-finetuning BERT on this task

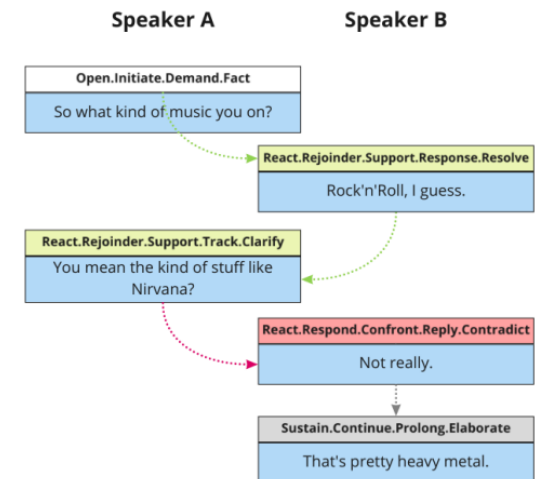
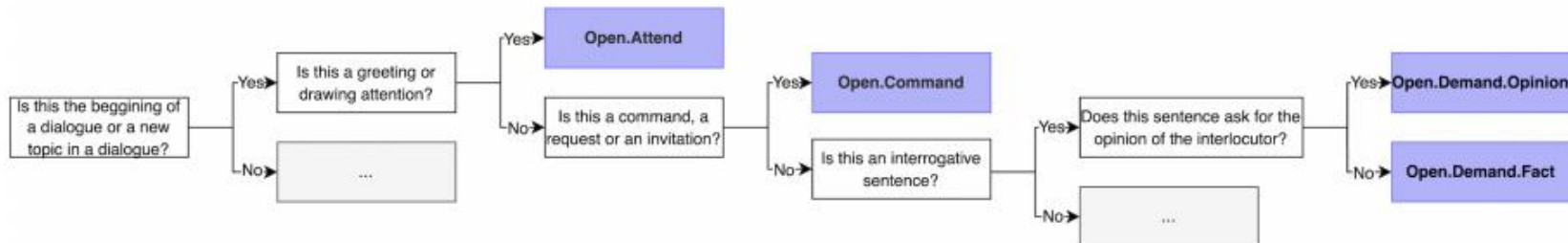
(Chen et al., 2019)
<http://arxiv.org/abs/1902.10909>



(Wu et al., 2020)
<https://www.aclanthology.org/2020.emnlp-main.66>

QA-style NLU

- Finetune PLM / use prompted LLM
- Ask specific questions relating to slots/intents
- Generate/copy reply
- PLMs: domain transfer
- LLMs: can be open-domain



(Ostyakova et al., 2023) <https://aclanthology.org/2023.sigdial-1.23>

Dialogue Management

- Given NLU input & dialogue so far, responsible for **deciding on next action**
 - keeps track of what has been said in the dialogue
 - keeps track of user profile
 - interacts with backend (database, internet services)
- Dialogue so far = **dialogue history**, modelled by **dialogue state**
 - managed by **dialogue state tracker**
- System actions decided by **dialogue policy**

Dialogue state / State tracking

- Stores (a summary of) dialogue history
 - User requests + information they provided so far
 - Information requested & provided by the system
 - User preferences
- Implementation
 - **handcrafted** – e.g. replace value for slot with last-mentioned
 - good enough in some circumstances
 - **probabilistic (belief state)**
 - keep an estimate of per-slot preferences based on NLU
 - more robust, more complex
 - accumulates probability over time & n-best lists
 - → handles NLU/ASR errors
 - e.g. 3x same low-confidence input = prob. high enough to react

price: cheap
food: Chinese
area: riverside

price: 0.8 cheap
0.1 moderate
0.1 <null>
food: 0.7 Chinese
0.3 Vietnamese
area: 0.5 riverside
0.3 <null>
0.2 city center

Basic State/Belief Trackers

a) Always trust the NLU

for **null** value:

$$p = \text{prev} \cdot p(\text{😬}) \sim \text{user didn't mention this slot}$$

non-null value v :

$$p = \text{prev} \cdot p(\text{😬}) + p(v)$$

~ didn't mention = carry from previous

~ did mention = add new NLU probability

- basically rule-based (but good if NLU is good)

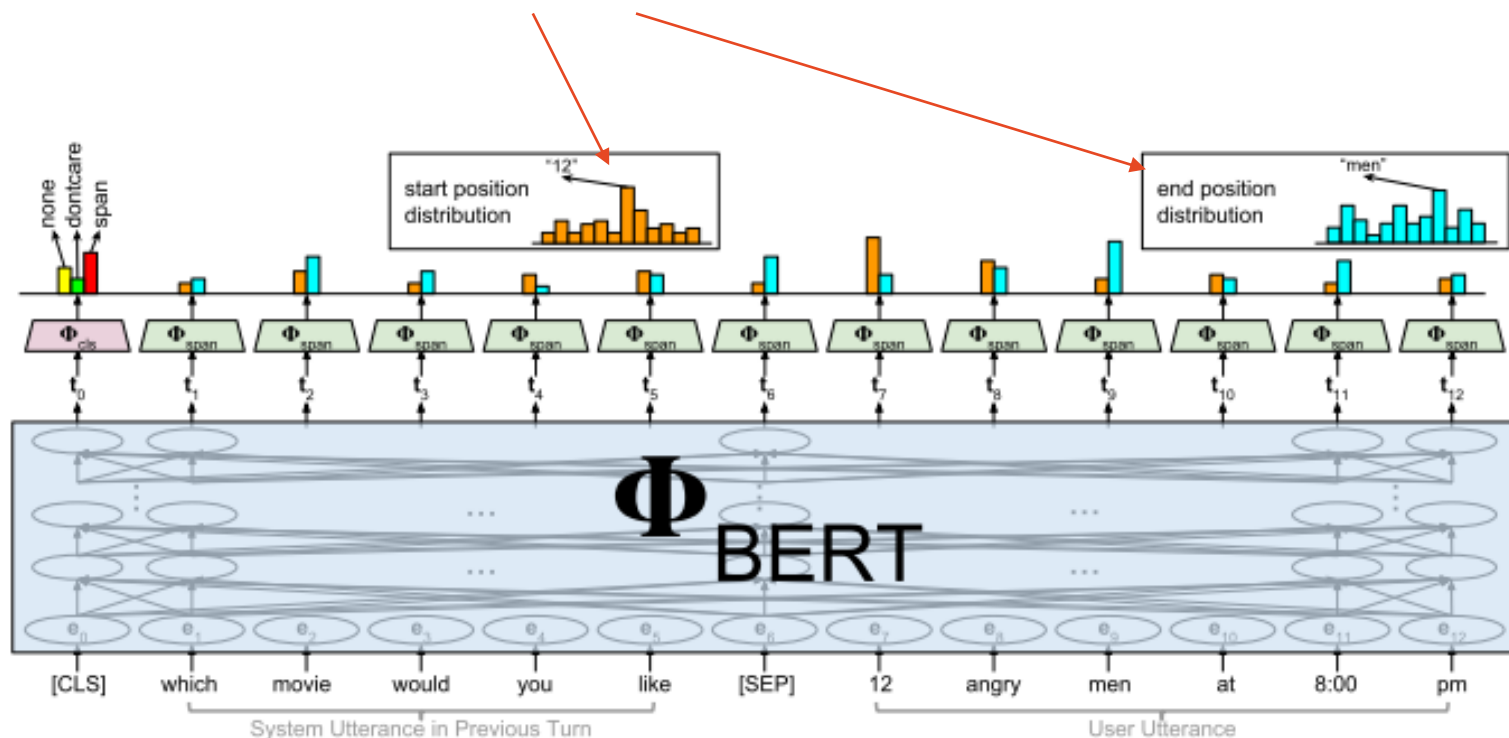
b) “NLU” over whole dialogue

- typically classification (“is slot value v present?”)
 - option: limit to some candidates (from NLU/delexicalization), rank them
- may not need NLU, may be better, but slower

State tracking by span tagging

- BERT over previous system & current user utterance
- from 1st token's representation, get a **decision**: *none*/*dontcare*/*span*
 - per-slot (BERT is shared, but the final decision is slot-specific)
- span = need to find a concrete value as a span somewhere in the text
 - **predict start & end token** of the span using 2 softmaxes over tokens
- rule-based update:
 - if *none* is predicted, keep previous value
 - essentially similar to NLU & update rule

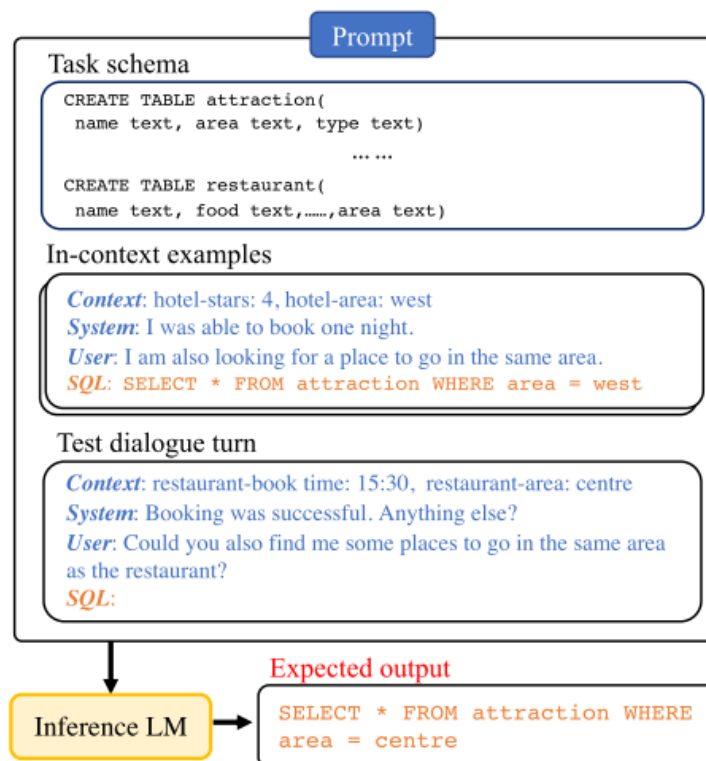
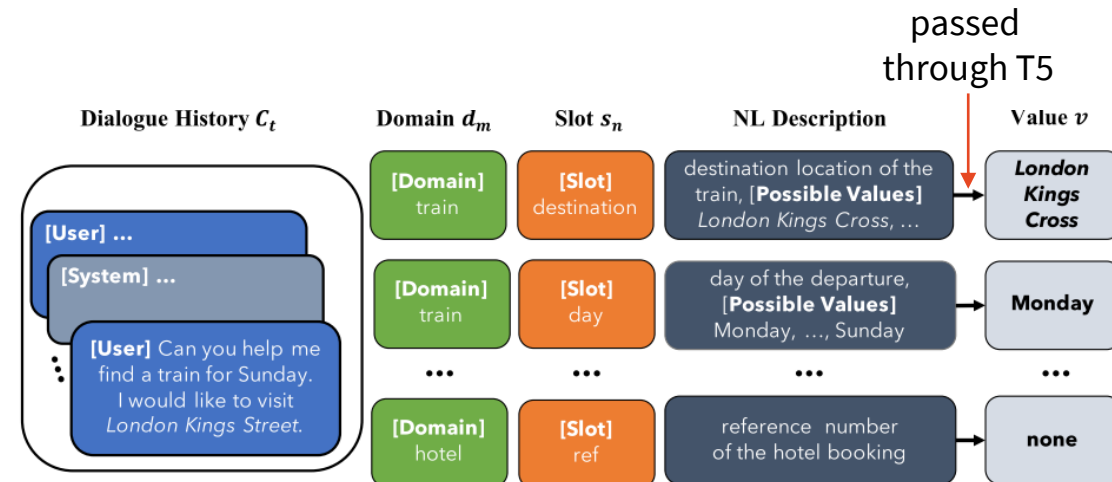
(Chao & Lane, 2019)
<http://arxiv.org/abs/1907.03040>



State tracking by generation

(Lee et al., 2021)
<https://aclanthology.org/2021.emnlp-main.404/>

- Finetune a PLM (T5) with specific prompts
 - dialogue history
 - domain + slot
 - (optional) slot description (+possible values)
- Generate values, not just selection
- LLMs: just prompt
 - make use of SQL for state



(Hu et al., 2022)
<https://aclanthology.org/2022.findings-emnlp.193>

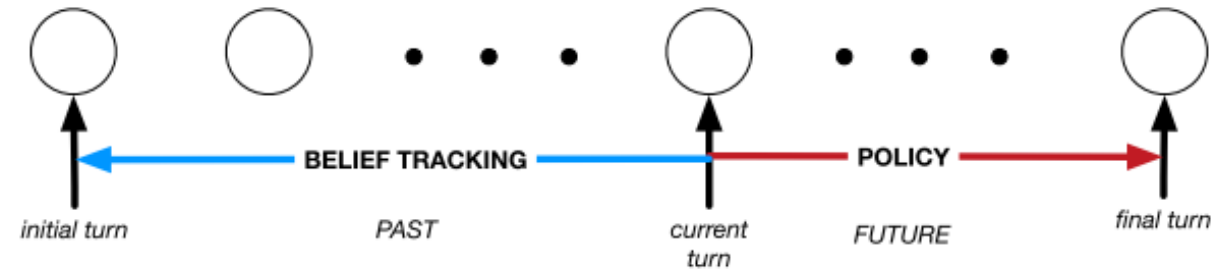
Action Selection / Policy

- **Deciding what to do next**

- **action** based on the current belief state
- following a **policy** (strategy) towards an end **goal** (e.g. book a flight)
- controlling the coherence & flow of the dialogue
- actions: linguistic & non-linguistic (backend access)
- actions represented by system dialogue acts

- DM/policy should:

- manage uncertainty from belief state ← *Did you say Indian or Italian?*
- recognize & follow dialogue structure ← follow convention, don't be repetitive
- plan actions ahead towards the goal ← e.g. ask for all information you require



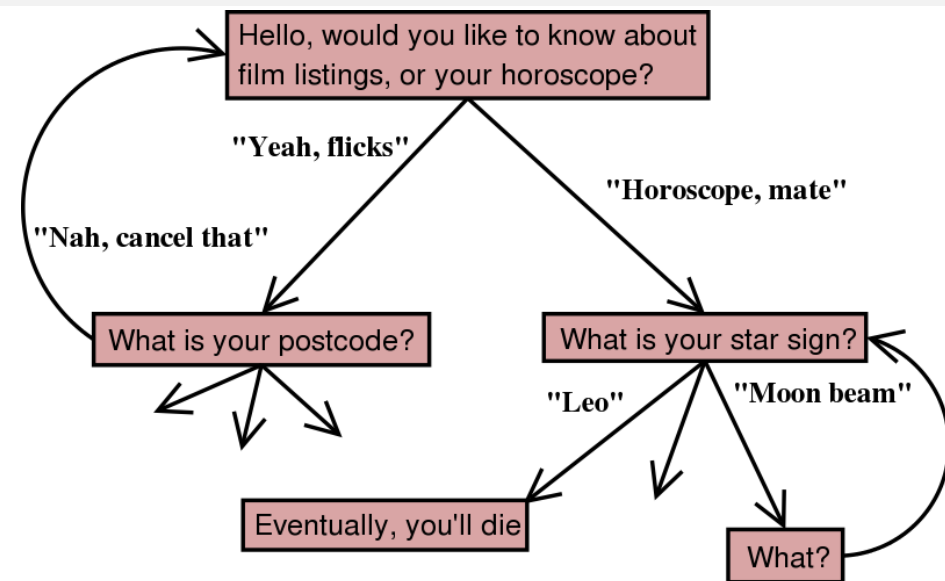
(from Milica Gašić's slides)

confirm(food=Chinese)

*inform(name=Golden Dragon,
food=Chinese, price=cheap)*

Action Selection Approaches

- Finite-state machines
 - simplest possible
 - dialogue state is machine state
- Frame-based/flowcharts (e.g. VoiceXML)
 - slot-filling + providing information – basic agenda
 - rule-based in essence
- Rule-based
 - any kind of rules (e.g. Python code)
- **Statistical – reinforcement learning**
 - allows more exploration than static data
 - allows more correct actions at a given point
 - plans ahead



```
229 elif fact['we_did_not_understand']:
230     # NLG("Sorry, I did not understand")
231     res_da = DialogueAct("notunderstood")
232     res_da.extend(self.get_limited_context())
233     dialogue_state["ludait"].reset()
234
235 elif fact['user_wants_help']:
236     # NLG("Pomoc.")
237     res_da = DialogueAct("help()")
238     dialogue_state["ludait"].reset()
239
240 elif fact['user_thanked']:
241     # NLG("Díky.")
242     res_da = DialogueAct('inform(cordina)')
243     dialogue_state["ludait"].reset()
244
245 elif fact['user_wants_restart']:
246     # NLG("Dobře, začneme znovu.")
```

(Dušek et al., 2014)

<https://www.aclweb.org/anthology/W14-4311>

RL for Dialogue Policy

(Li et al., 2017)

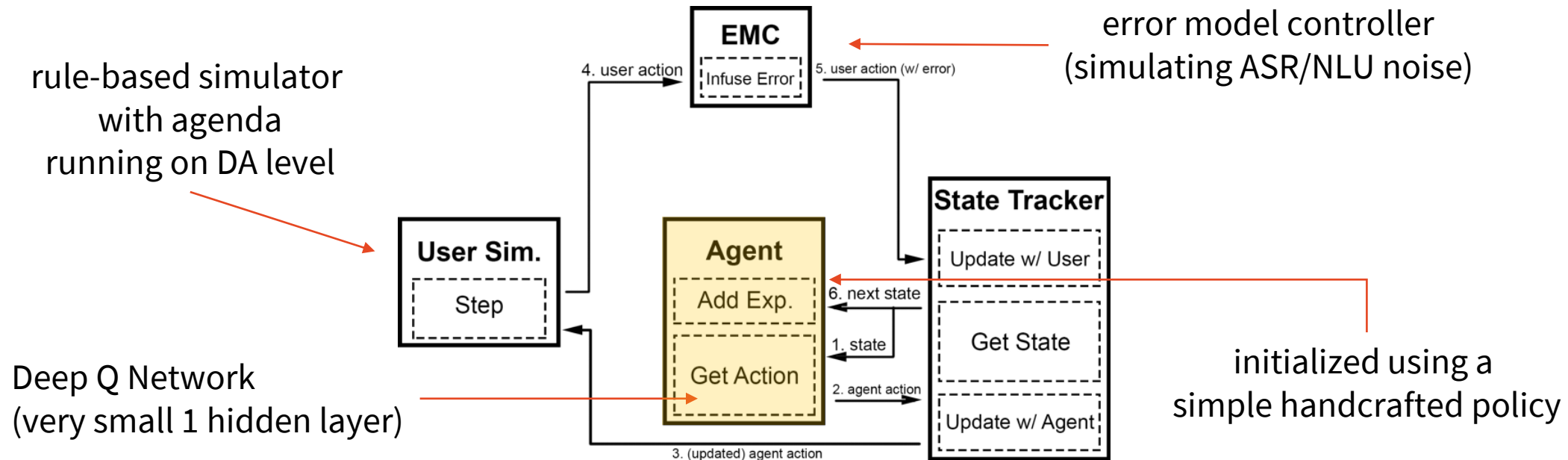
<https://arxiv.org/abs/1703.01008>

<https://github.com/MiuLab/TC-Bot>

(Lipton et al., 2018)

<https://arxiv.org/abs/1608.05081>

- Directly use a RL algorithm to drive dialogue action selection/policy
 - Deep Q Networks: estimate state values
 - Policy Gradients : directly optimize policy
- Needs to use a simulator ~ another dialogue system
 - would be too expensive & slow & tedious with people



Natural Language Generation (NLG) / Response Generation

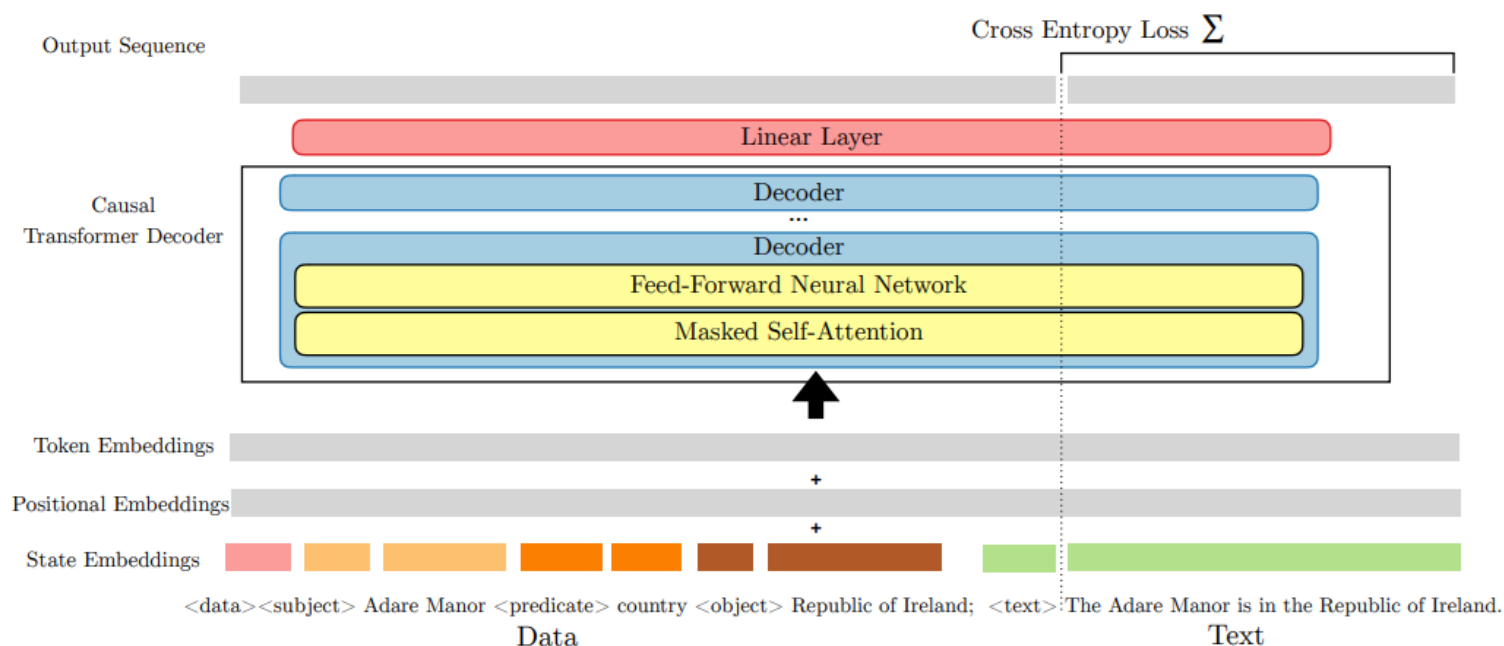
- Representing system dialogue act in natural language (text)
 - reverse NLU
- How to express things might depend on context
 - Goals: fluency, naturalness, avoid repetition (...)
- Traditional approach: **templates**
 - Fill in (=lexicalize) values into predefined templates (sentence skeletons)
 - Works well for limited domains

inform(name=Golden Dragon, food=Chinese, price=cheap)
+
<name> is a <price>-ly priced restaurant serving <food> food
=
Golden Dragon is a cheaply priced restaurant serving Chinese food.

- Statistical approach: **seq2seq**/language models
 - input: system dialogue act, output: sentence

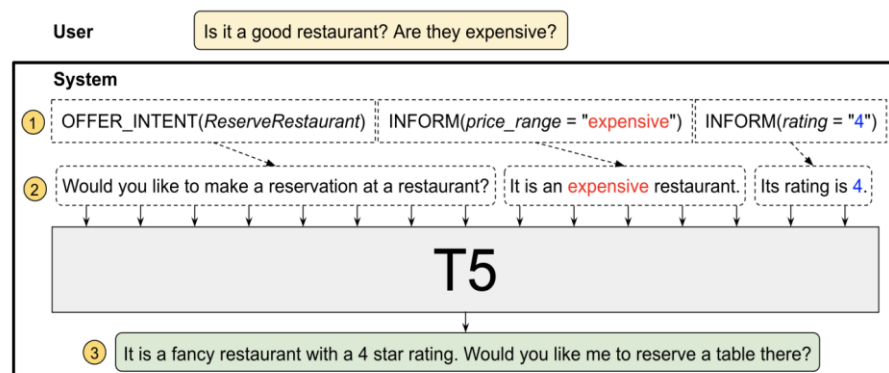
- Finetuning Transformer-based PLMs
- input: structured meaning representation, output: text
- PLMs good at implicit copying, can use multilingual models
- Improvement: overgenerate & rerank
 - another PLM to check, pick output with fewest errors

(Harkous et al., 2020)
<http://arxiv.org/abs/2004.06577>



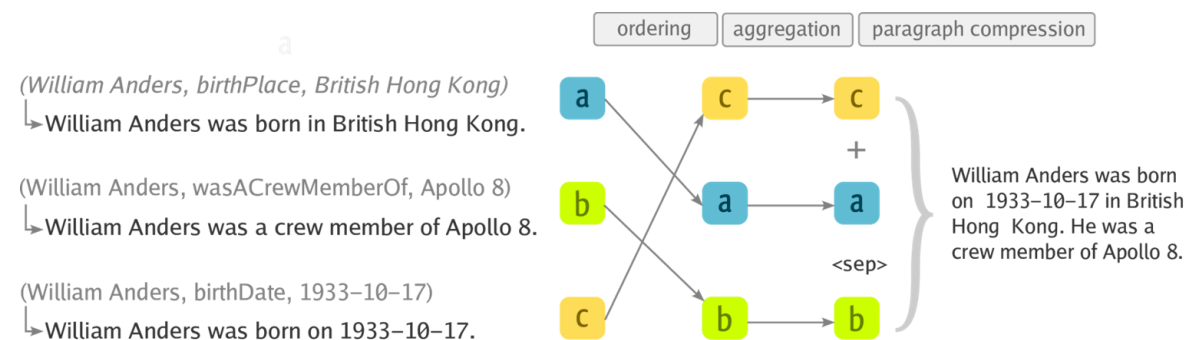
Finetuned PLMs with Templates

- goal: improving semantic accuracy + controllability
- concatenate simple templates, then use PLMs to make text more fluent
- advantages of templates (controllability) + neural LMs (fluency)



(Kale & Rastogi, 2020)

<https://www.aclweb.org/anthology/2020.emnlp-main.527>



(Kasner & Dušek, 2022)

<https://aclanthology.org/2022.acl-long.271/>

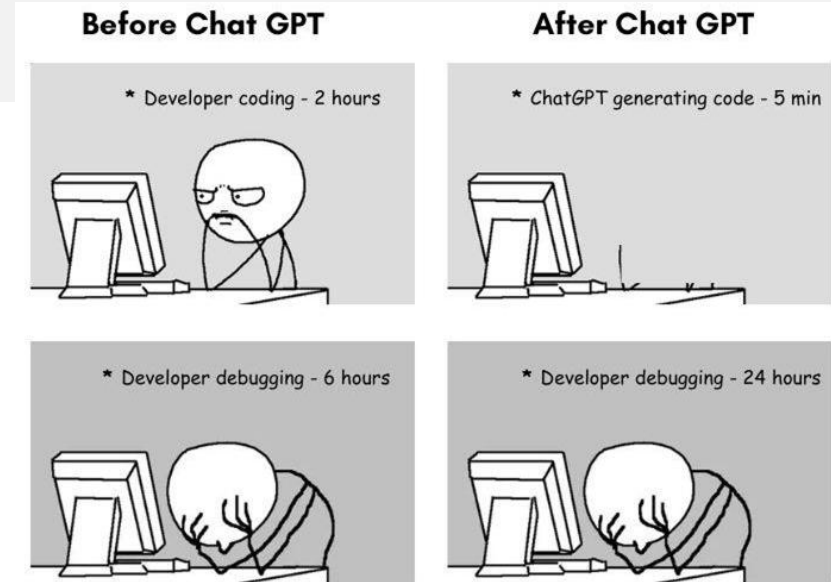
(PLMs can be also replaced with a prompted LLM)

(Xiang et al., 2022)

<http://arxiv.org/abs/2210.04325>

- prompting instead of task-specific finetuning
 - LLMs work with common data formats (JSON, CSV...)
- for NLG, prompting competitive to finetuning, but has other problems:
 - variability in responses

*Here is the answer: (...),
As an AI language model (...)*
 - prompt sensitivity
 - semantic errors
- issues with closed models:
replicability, cost, data contamination



Prompt

```
Based on the given data:
```
{data}
```

Your task is to write a brief, fluent,
and coherent single-paragraph {output_type}
in natural language. The text should be
balanced and neutral. Make sure that all the
facts mentioned in the text can be derived
from the input data, do *not* add any extra
information.
```

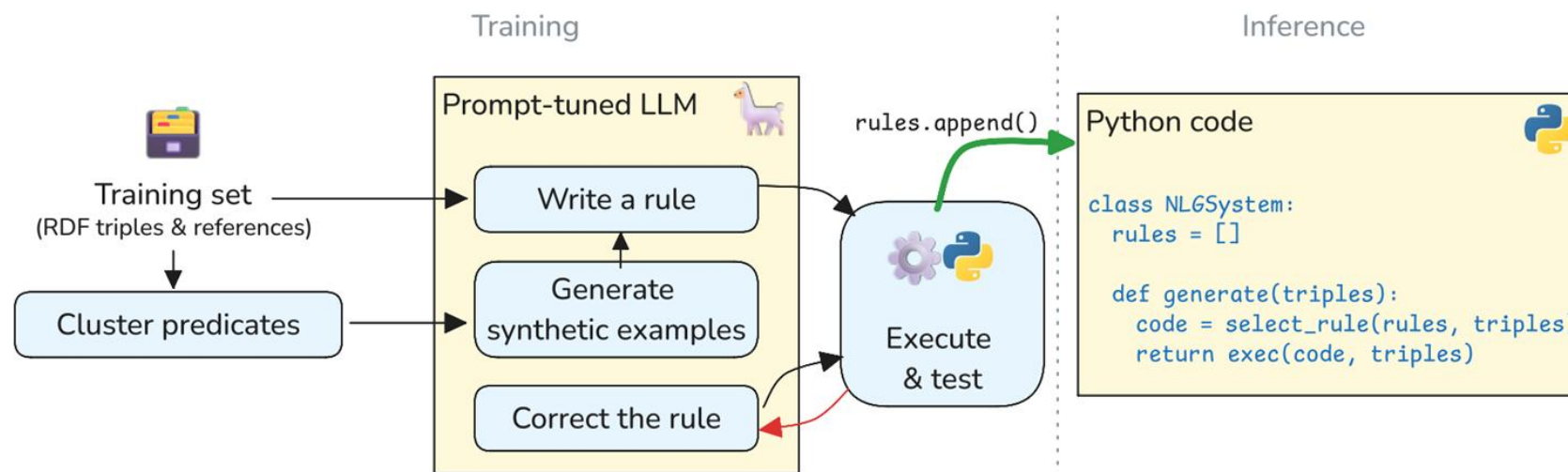
Output prefix

```
Sure! Here is the {output_type}:
"
```

Rule-based NLG system with LLMs

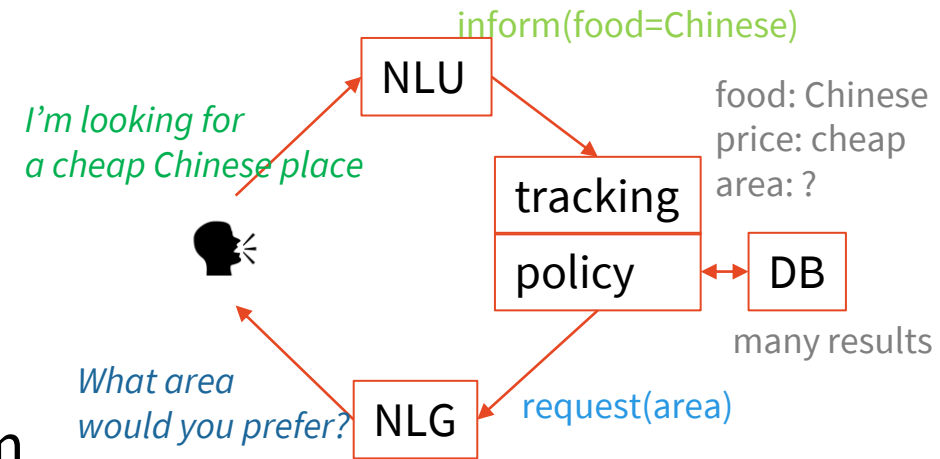
(Warczynski et al., 2024)
<https://aclanthology.org/2024.inlg-main.48>

- instead of generating the output directly, we can use **LLMs to generate the rules**
 - e.g. Python string templates
- fewer hallucinations, CPU-only inference
- only limited generalization to unseen data



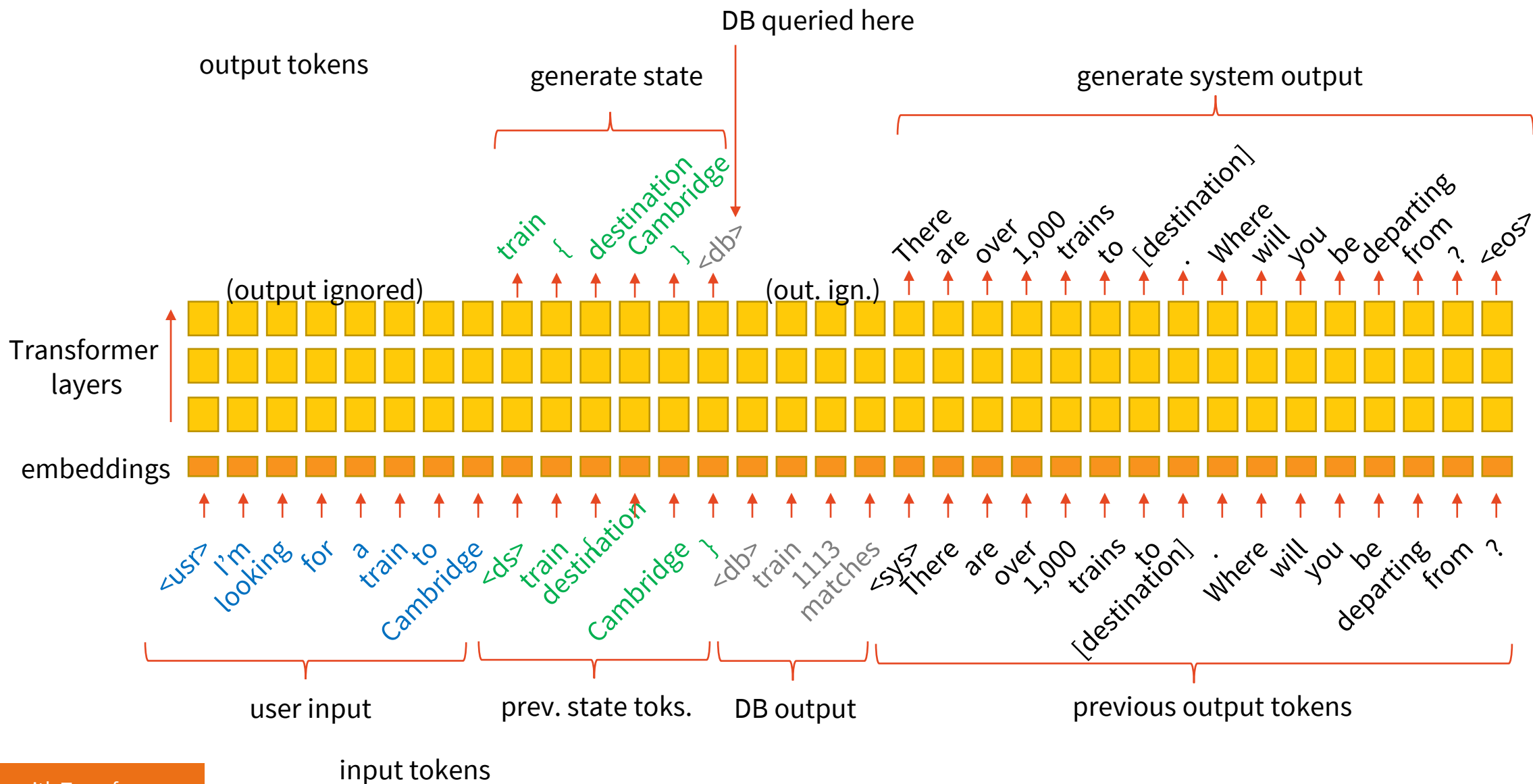
End-to-end models (vs. separate components)

- Separate components (NLU→DM→NLG):
 - more flexible (replace one, keep the rest)
 - more explainable
 - trained separately, possibly optimization by RL
 - error accumulation
 - improved components don't mean improved system
- **End-to-end models:**
 - single neural network for NLU&DM&NLG
 - joint supervised optimization, RL still works
 - PLMs need a lot of finetuning data
 - less control of outputs: hallucination, dull/repetitive



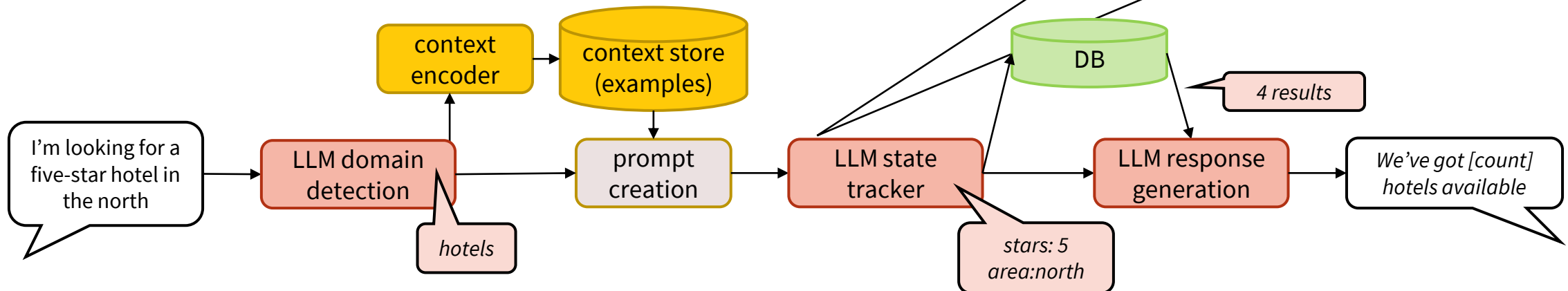
A handwritten example of an end-to-end model output. The input is "How are you <sb>". The output is "I am good". The model is labeled "NN" (Neural Network). Red arrows point from the input tokens to the output tokens, indicating the model's internal processing.

End-to-end Task-oriented Dialogue with GPT-2



LLMs for Task-Oriented Dialogue

- Prompting LLMs for the task
 - still: input → **state** → **DB** → **response**
 - additional 1st step: **domain detection**
 - tracking & response prompts domain specific
- Zero or few-shot setting
 - few-shot examples/snippets help
 - little or no data needed: wide potential



instruction

domain description

examples

dial. history

user input

Definition: Capture values from a conversation about hotels. Capture pairs “entity:value” separated by colon and no spaces in between. Separate the “entity:value” pairs by hyphens. Values that should be captured are:

- “pricerange”: the price of the hotel
- “area”: the location of the hotel

...

--- Example 1 ---

...

...

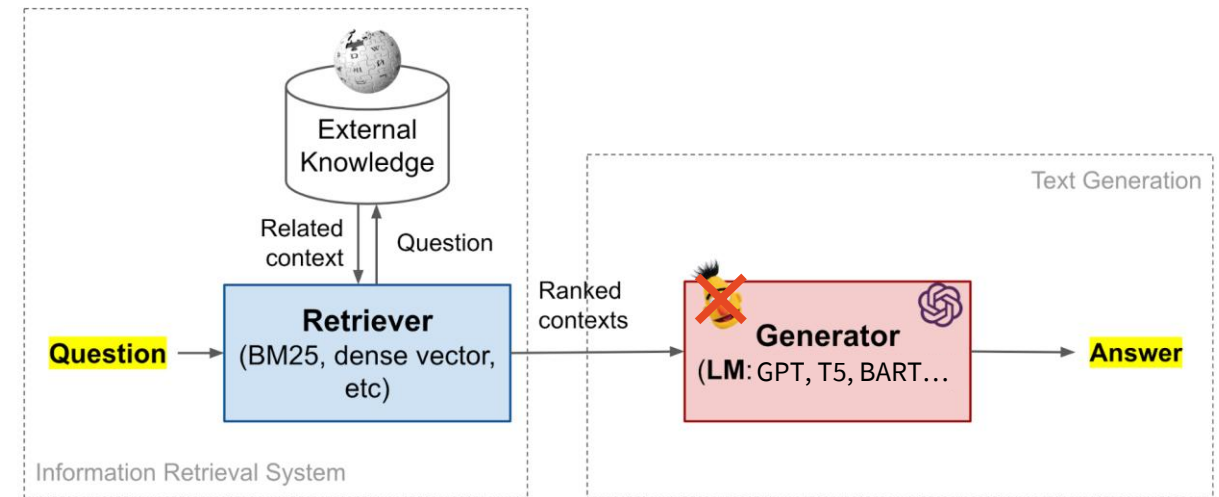
Assistant: “Hello, how can I help you?”

...

Customer: “I am looking for a five-star hotel in the north”

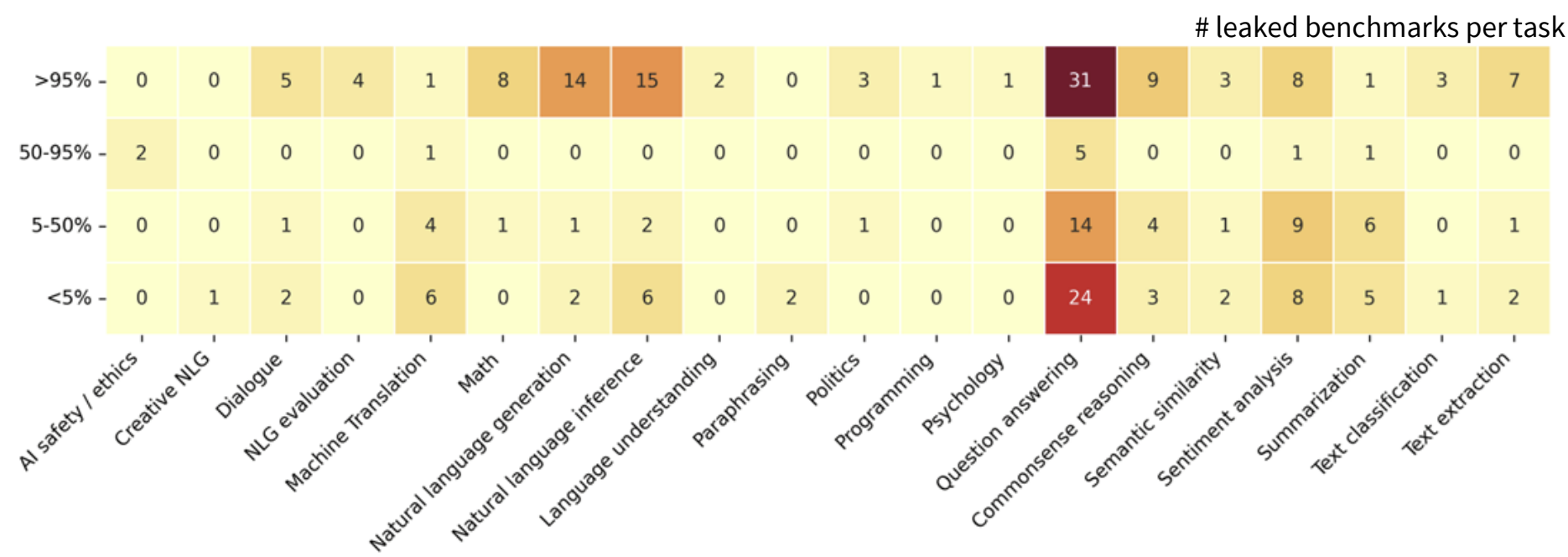
Retrieval-augmented Generation

- Easiest way to employ LLMs in dialogue
 - often deployed in production now
 - Less reliable, less structured
- Feed any data to LLM, then ask
 - reduce LLM hallucination
 - doesn't need structure
 - no intents & state
 - LLMs are good enough to not just copy
- Retrieval is kinda optional
 - LLM context windows are 100k's of tokens – might be enough for all data
 - still might improve behavior
- Options: ask LM if the retrieved is relevant, ask to link to sources

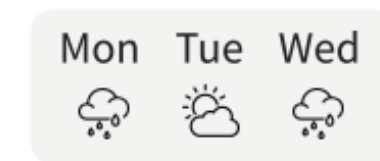


<https://lilianweng.github.io/posts/2020-10-29-odqa/>

- Saturated: models constantly tested on them
 - MMLU, GSM8K – also, not that much generation
- Contaminated: data leaked to model training (closed LLMs, open-weight LLMs)
 - not just pretraining: indirect leakage to closed LLMs
 - >200 benchmarks leaked to ChatGPT via web interface



- Automatic metrics (e.g. BLEU): text → score (number)
BLEU = **0.184**
- Trained metrics
BertScore = **0.916**
- LLM scores
I'd rate the factual accuracy as a **3 out of 10**.
- Humans: Y/N, Likert scales
Is the output factually accurate? Yes / **No**



LLM

Skies will be mostly clear, but winds will remain strong.

- Good but comparison only, no deeper insight
 - we don't know why the score is low/high

reference:

Skies will be mostly cloudy with rain showers, but it will clear up on Tuesday.

- Better option:
find individual errors – mark error spans

- more detailed feedback, actionable
- still allows comparing models

incorrect
↓
Skies will be mostly clear, but
winds will remain strong.
↑
not checkable

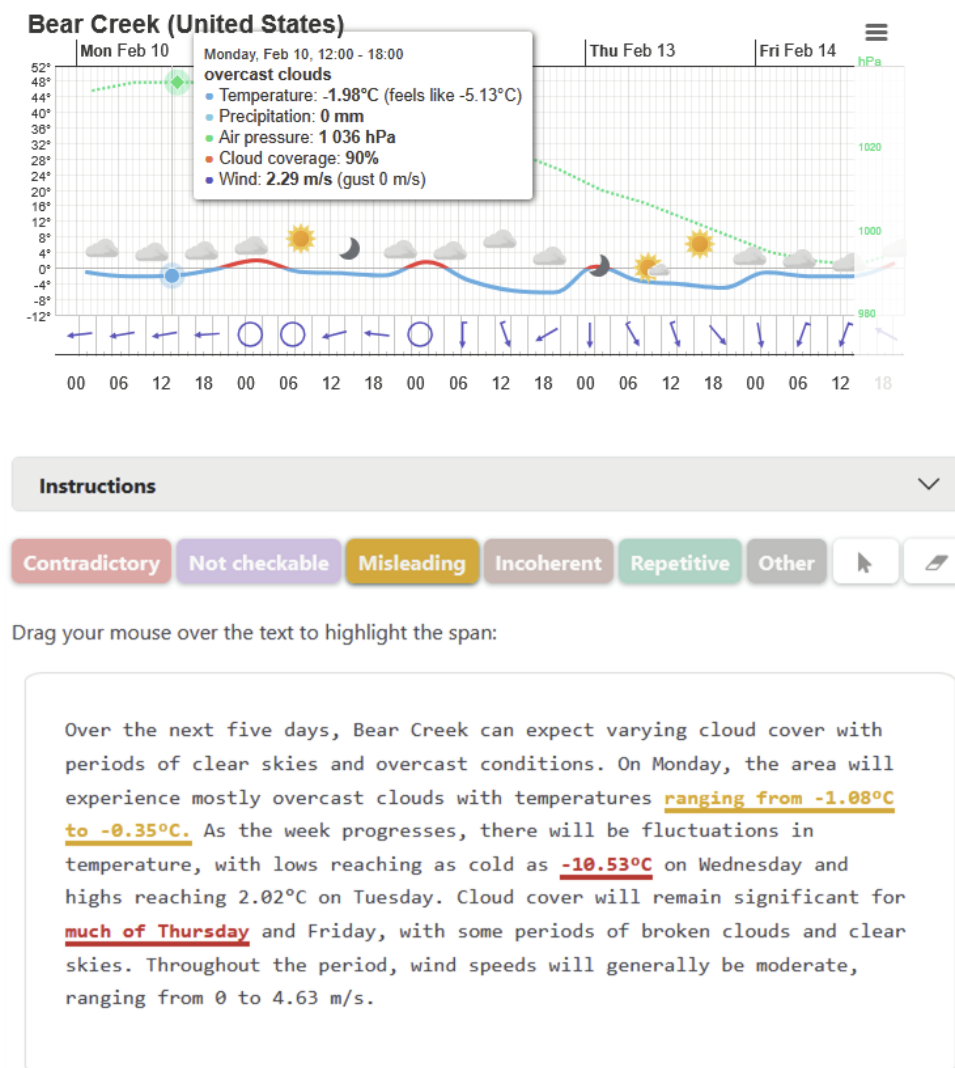
Factgenie: Simplifying evaluation

<https://github.com/ufal/factgenie>
(Kasner et al., 2024)
<https://aclanthology.org/2024.inlg-demos.5>

- A tool for span-based evaluation
 - data visualization
 - annotation
 - analysis
- Various input data formats
 - JSON, CSV, HTML
 - custom (e.g. weather)
- Use humans & LLMs in comparable setups
- NLG on unseen open data with ~7B models:
80% outputs has errors

(Kasner & Dušek, 2024)

<https://aclanthology.org/2024.acl-long.651>



- Multi-aspect span annotation with LLMs
 - Ensemble of open LLMs
 - Detailed aspect-specific prompts
- Find spans & explain + provide overall score & explain
- Comparable to closed LLMs

Context: [SUMMARY OF THE STORY]

Question: What is Dr. Heywood Floyd’s mission on the Clavius Base?

Answer: According to the summary provided, Dr. Heywood Floyd’s mission on the Clavius Base is to investigate a recently found artifact buried four million years ago. Specifically, his mission is to ride in a Moonbus to the artifact, which is a monolith identical to the one encountered by the man-apes in the summary.

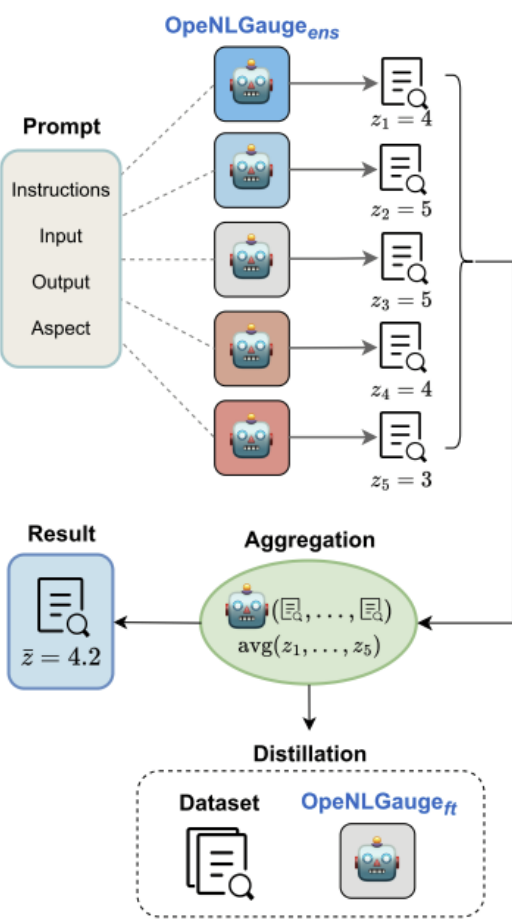
Evaluation Aspect: Conciseness – Extent to which the answer is concise and to the point

Explanation: This introductory phrase is an error affecting conciseness as it adds unnecessary words to the answer. The response could directly state Floyd’s mission without referencing the source (summary), given the context of the question.
Severity: 2

Explanation: This phrase is considered an error affecting conciseness because it provides additional, unnecessary details about the mission. The initial sentence already clearly states Floyd’s mission, making the subsequent elaboration redundant and wordy.
Severity: 3

Overall Score: Fair (3/5)

Explanation of the Score: While the generated answer generally addresses the question, its conciseness is compromised by the inclusion of redundant information and an unnecessary introductory phrase. Removing these elements would enhance the answer’s directness and efficiency, potentially elevating the score to ‘Good’.



Conclusion & Further

We have useful systems, but the tasks are far from solved

- reliability (~hallucinations) are arguably the biggest issue

Other open research questions

(Defossez et al., 2024) <https://arxiv.org/abs/2410.00037>

(Ji et al., 2024) <http://arxiv.org/abs/2411.13577>

- End-to-end audio models: fully audio based models in a single LLM, low latency
- Incrementality: don't wait for the whole sentence to start processing
- Flow induction: analyze call center recordings automatically → build chatbot
- Context dependency: understand/reply in context (grounding, speaker alignment)

(Zhou et al., 2022) <https://aclanthology.org/2022.acl-long.110>

(Burdisso et al., 2024) <https://aclanthology.org/2024.emnlp-main.310/>

(Kumar & Dušek, 2024) <https://aclanthology.org/2024.findings-naacl.46>

Thanks

Contacts:

Ondřej Dušek

odusek@ufal.mff.cuni.cz

<https://tuetschek.github.io>

[@tuetschek](#)

Thanks to Zdeněk Kasner (<https://kasnerz.github.io/>) for some of these slides

