NPFL099 Statistical Dialogue Systems

8. Natural Language Generation

Zdeněk Kasner, Ondřej Dušek, Mateusz Lango, Ondřej Plátek

http://ufal.cz/npfl099

21.11.2024

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

Natural Language Generation

= task of automatically producing text in e.g. English (or any other language)

- covers many subtasks:

task	input	output
machine translation	*text in language A*	*text in language B*
summarization	*long text*	*text summary*
question answering	*question*	*answer*
image captioning	*image*	*image caption*
story generation	*topic*	*story*
paraphrasing	*text*	*paraphrased text*
data-to-text generation	*structured data*	*description of the data*
dialogue response generation	*dialogue act*	*system response*

NLG in a narrow sense

NLG Objectives

- general NLG objective:

> given **input & communication goal**
>
> create **accurate + natural, well-formed, human-like text**

- additional NLG desired properties:
  - variation (avoiding repetitiveness)
  - simplicity (saying only what is intended)
  - adaptability (conditioning on e.g. user model)

NLG in Dialogue Systems

- in the context of dialogue systems:

NLG: **system action → system response**

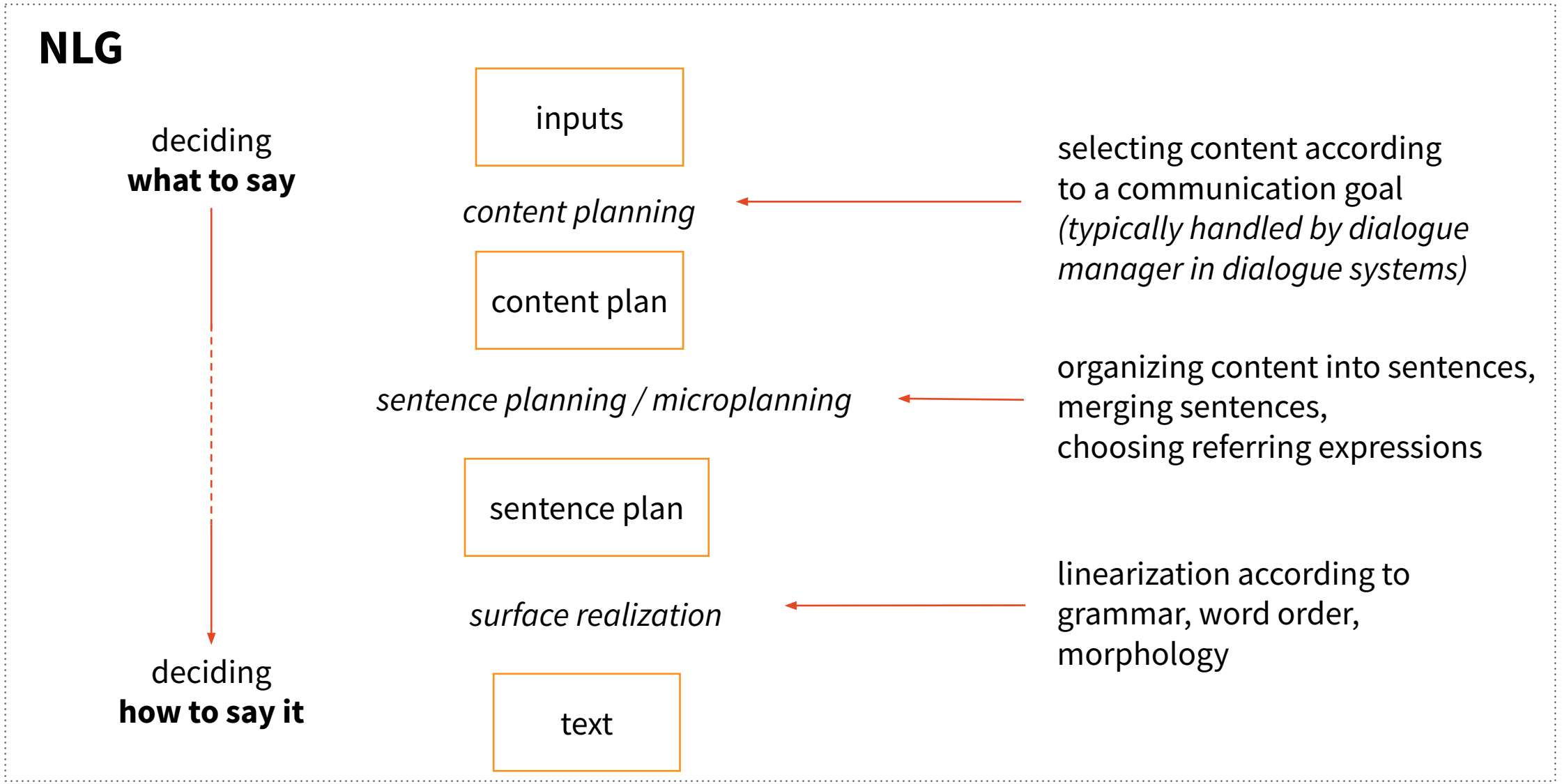
"what the system wants to say"

"actually saying it"

- system action
  - selected by the dialogue manager
  - may be conditioned on:
    - dialogue state
    - dialogue history (→ referring expressions, avoiding repetition)
    - user model (→ "user wants short answers")

NLG Subtasks (Textbook Pipeline)

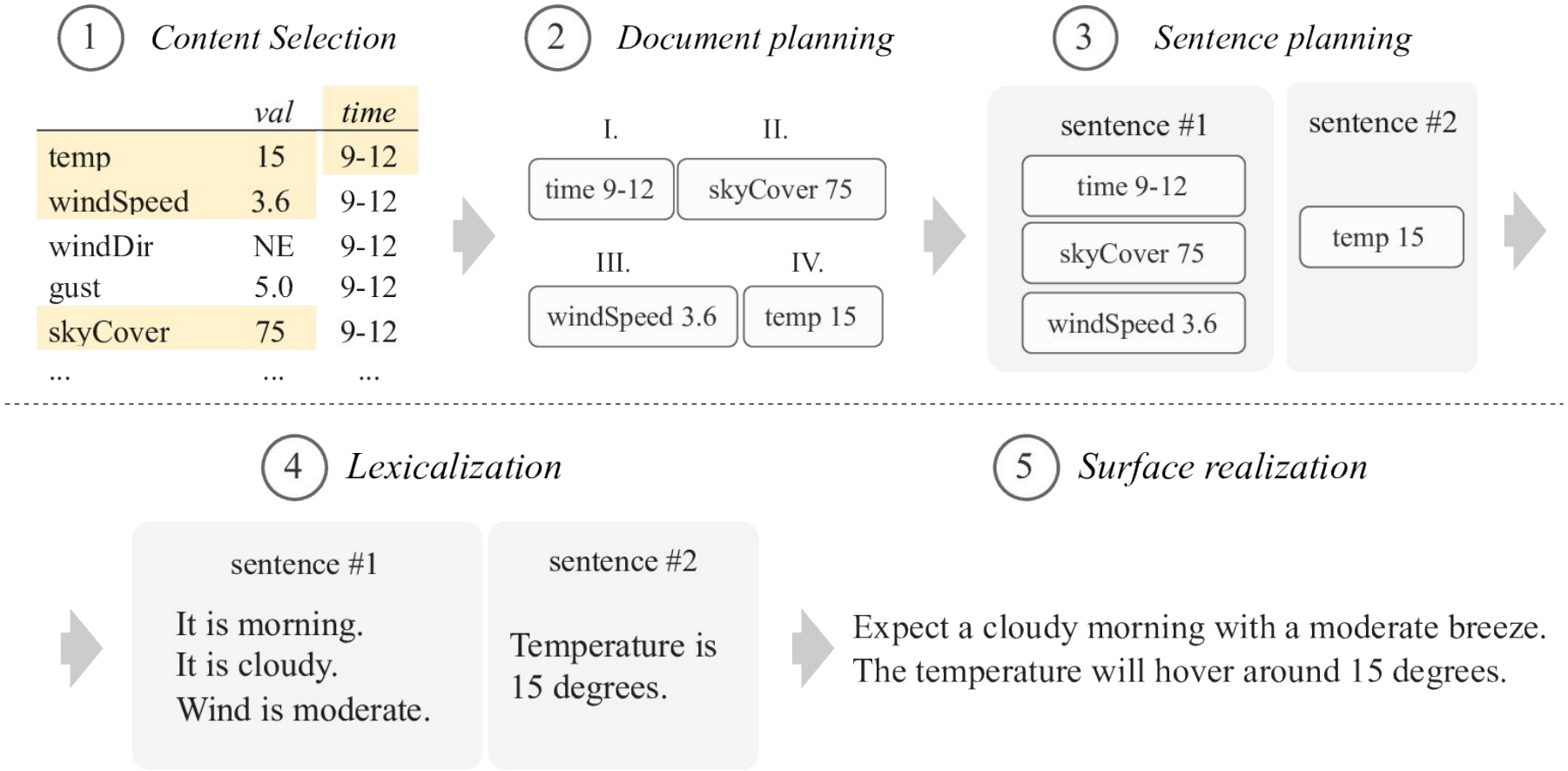
= how proper NLG had to be done before neural approaches



NLG Subtasks (Textbook Pipeline)

(Zdeněk K.'s PhD Thesis, Figure 2.4)
https://dspace.cuni.cz/handle/20.500.11956/193018

Example: classical NLG pipeline

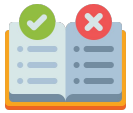


NLG Basic Approaches

- **hand-written prompts** *(“canned text”)*
  - most trivial – hard-coded, no variation
  - doesn’t scale (good for DTMF phone systems)

- **templates** *(“fill in blanks”)*
  - simple, but much more expressive – covers most common domains nicely
  - can scale if done right, still laborious
  - most production dialogue systems

- **grammars & rules**
  - grammars: mostly older research systems
  - rules: mostly content & sentence planning



- **machine learning**
  - modern research systems
  - pre-neural attempts often combined with rules/grammar
  - NNs made it work much better

Template-based NLG

- most common in commercial dialogue systems
- simple, straightforward, reliable
  - custom-tailored for the domain
  - complete control of the generated content
- lacks generality and variation
  - difficult to maintain, expensive to scale up
- can be enhanced with rules
  - e.g. articles, inflection of the filled-in phrases
  - template coverage/selection rules (heuristics, random variation)
- can be a good starting point for ML algorithms
  - post-editing / reranking the templates with neural language models

Template-based NLG – Examples

Example: Facebook

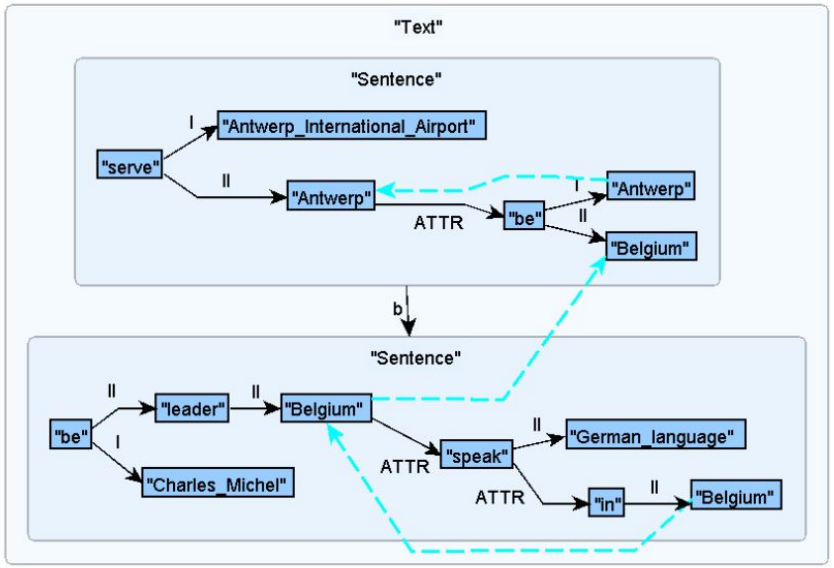
(Facebook, 2015)

(Facebook, 2019)

inflection rules

Grammar / Rule-based NLG

- based on top of linguistic theories
- state-of-the-art research systems until NLG the arrival of NNs
- rules for building tree-like structures
  → rules for tree linearization
- reliable, more natural than templates
- takes a lot of effort, naturalness still not human-level
- see NPFL123 for more details



(Mille et al., 2019)
https://aclanthology.org/W19-8659.pdf

Neural NLG

- learning the task from the data
- sequence-to-sequence generation
- fluency can match human-level, minimal hand-crafting
- not controllable (“black-box”),
  semantic inaccuracies (omissions / hallucinations),
  low diversity,
  expensive data gathering,
  expensive training,
  expensive deployment

  → promising research area 😉
- getting better with larger models

Seq2seq Generation

https://slds-lmu.github.io/seminar_nlp_ss20/natural-language-generation.html

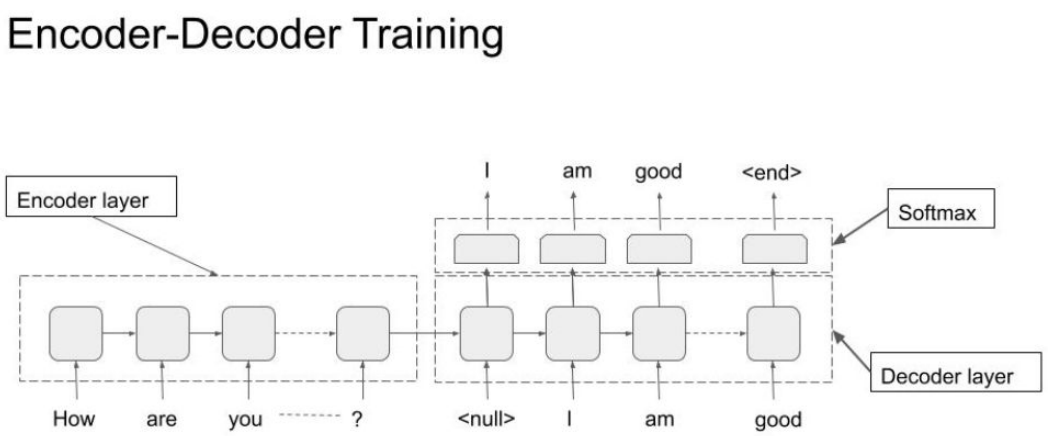
decoder-only

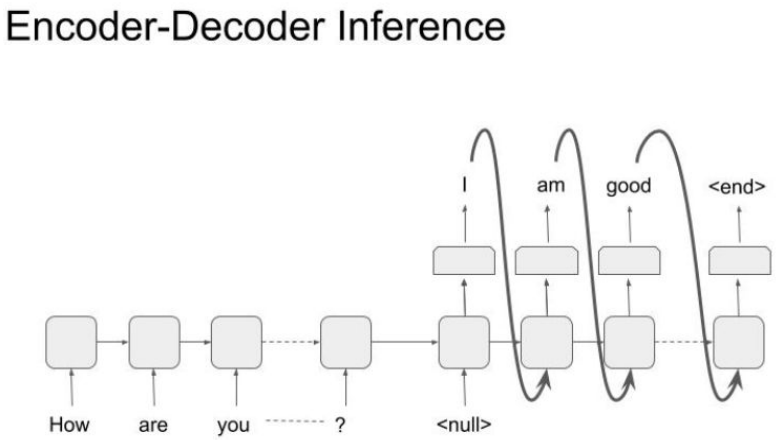
- input sequence is prepended as a context, the decoder generates continuation
- pretrained Transformers (PLMs): GPT-2, all LLMs

encoder-decoder

- *RNN:* encoder updates the hidden state → decoder is initialized with the hidden state
- *Transformer*: encoder generates a sequence of hidden states → decoder attends to this sequence
- PLMs: BART, T5

- training vs. inference:





Decoding Algorithms

- for each time step *t*, the decoder outputs a probability distribution: $P(y_t | y_{1:t-1}, \mathbf{X})$
- how to use it?
- **exact inference:** find a sequence maximizing $P(y_{1:T} | \mathbf{X})$
  - not possible in practice (why? and is it our goal?)
- **approximation algorithms**
  - greedy search
  - beam search
- **stochastic algorithms**
  - random sampling
  - top-k sampling
  - nucleus sampling (=top-p sampling)

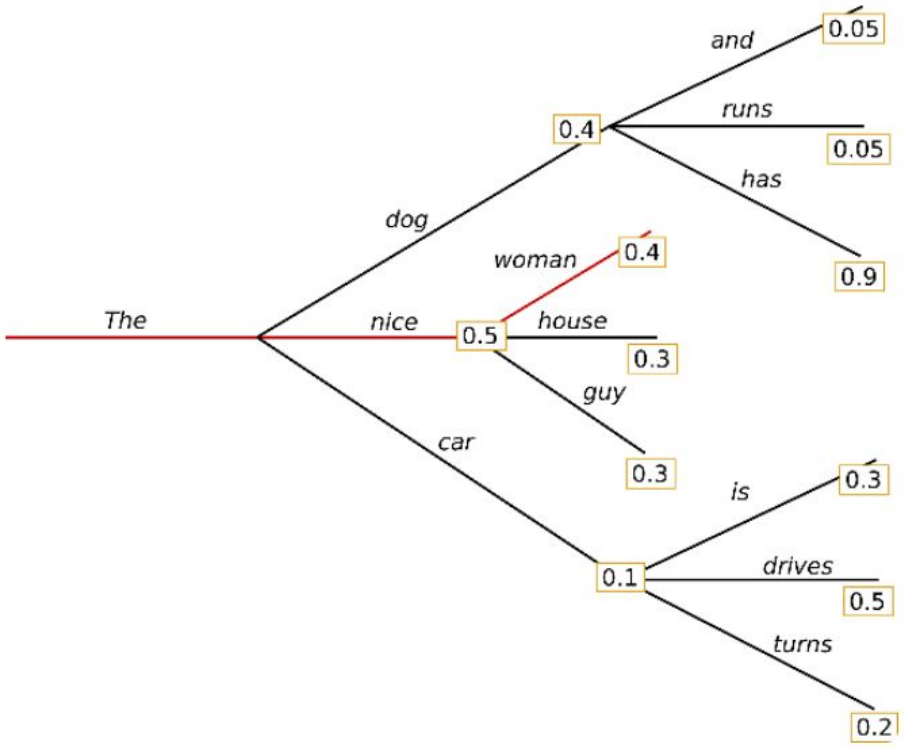
(+ repetition penalty → decreasing probabilities of generated tokens)

Decoding Algorithms

https://huggingface.co/blog/how-to-generate
https://towardsdatascience.com/decoding-strategies-that-you-need-to-know-for-response-generation-ba95ee0faadc

Greedy search: always take the argmax

- does not necessarily produce the most probable sequence (why?)
- often produces dull responses



Example:

Context:	Try this cake. I baked it myself.
Optimal Response :	This cake tastes great.
Greedy search:	This is okay.

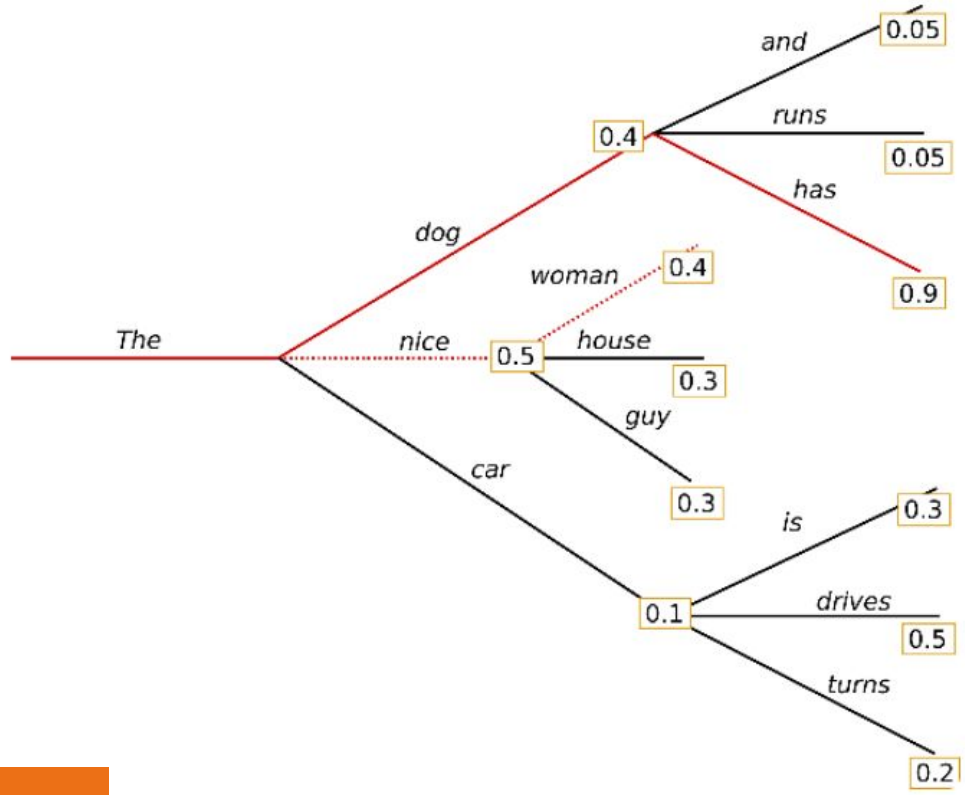
many examples start with “This is”,
no possibility to backtrack

Decoding Algorithms

https://huggingface.co/blog/how-to-generate

Beam search: try *k* continuations of *k* hypotheses, keep *k* best

- better approximation of the most probable sequence, bounded memory & time
- allows re-ranking generated outputs
- k=1 → greedy search



Reranking:

is there a later time

inform_no_match(alternative=next)

-2.914 No route found later , sorry .
-3.544 The next connection is not found .
-3.690 I'm sorry , I can not find a later ride .
-3.836 I can not find the next one sorry .
-4.003 I'm sorry , a later connection was not found .

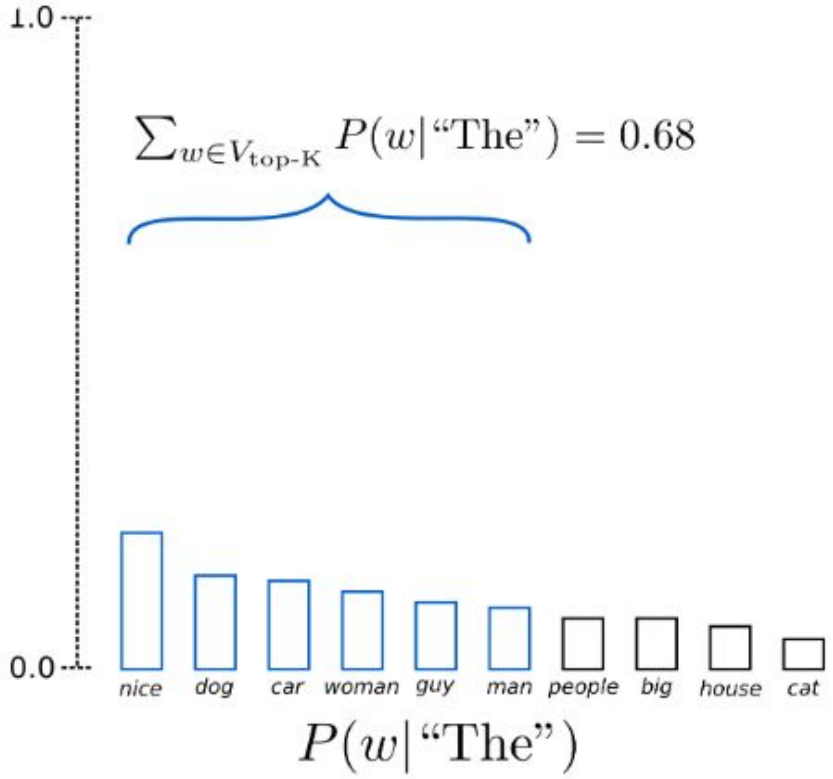
(Ondřej's PhD thesis, Fig. 7.7)
http://ufal.mff.cuni.cz/~odusek/2017/docs/thesis.print.pdf

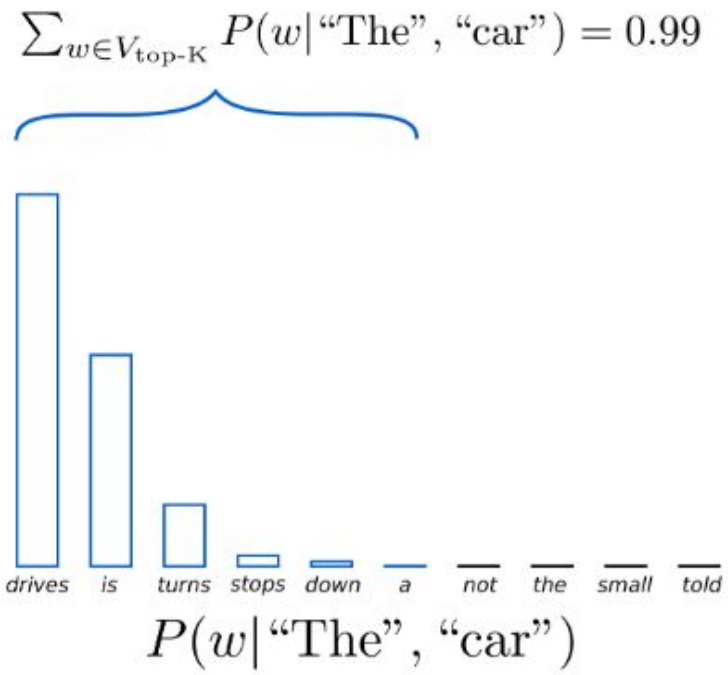
Decoding Algorithms

https://huggingface.co/blog/how-to-generate

Top-k sampling: choose top k options (~5-500), sample from them

- avoids the long tail of the distribution
- more diverse outputs



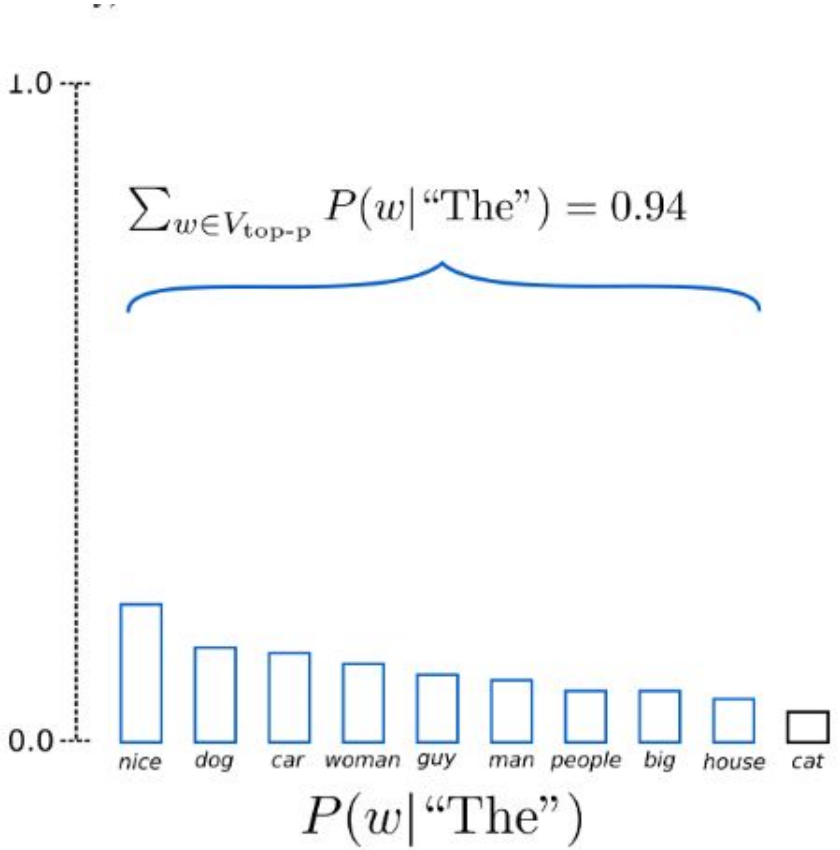


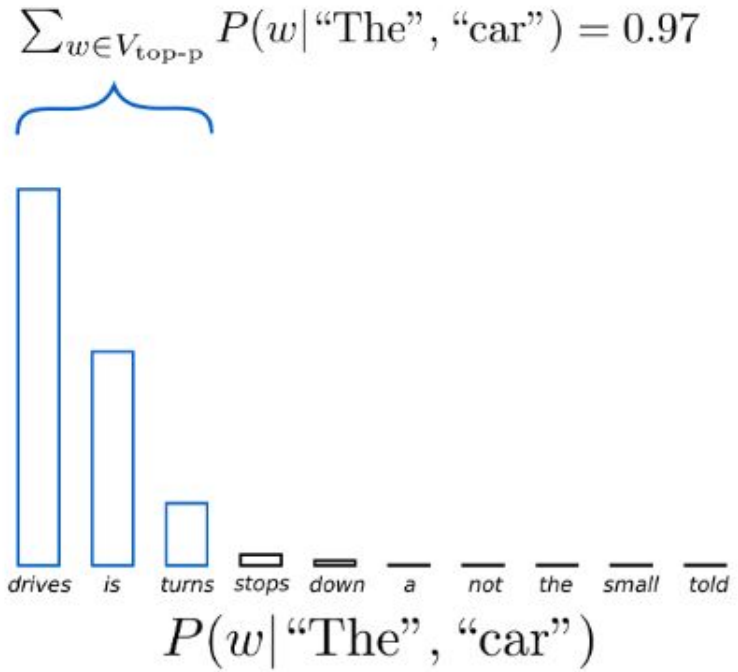
Decoding Algorithms

https://huggingface.co/blog/how-to-generate

Top-p (nucleus) sampling: choose top options that cover >= *p* probability mass (~0.9)

- can be viewed as "*k*" from top-k adapted according to the distribution shape





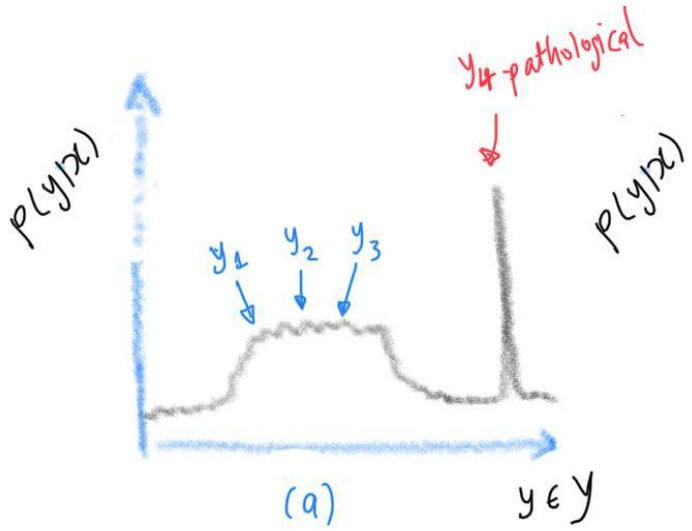
Decoding Algorithms

https://suzyahyah.github.io/bayesian%20inference/machine%20translation/2022/02/15/mbr-decoding.html

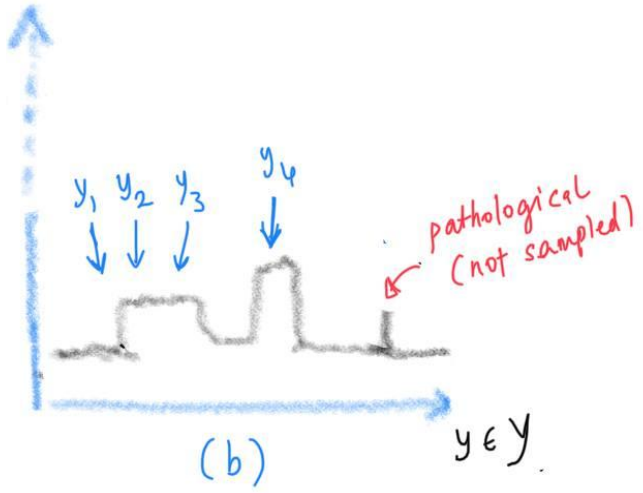
Minimum Bayes Risk (MBR): Selecting the sequence most similar to other sequences = “consensus decoding”

- useful for minimizing pathological behavior, e.g. decoding an empty sequence.
- intractable → we need a sampling algorithm
  - **epsilon sampling**: sampling only tokens with a probability larger than epsilon

(Freitag et al., 2023)
https://aclanthology.org/2023.findings-emnlp.617



MAP 😕

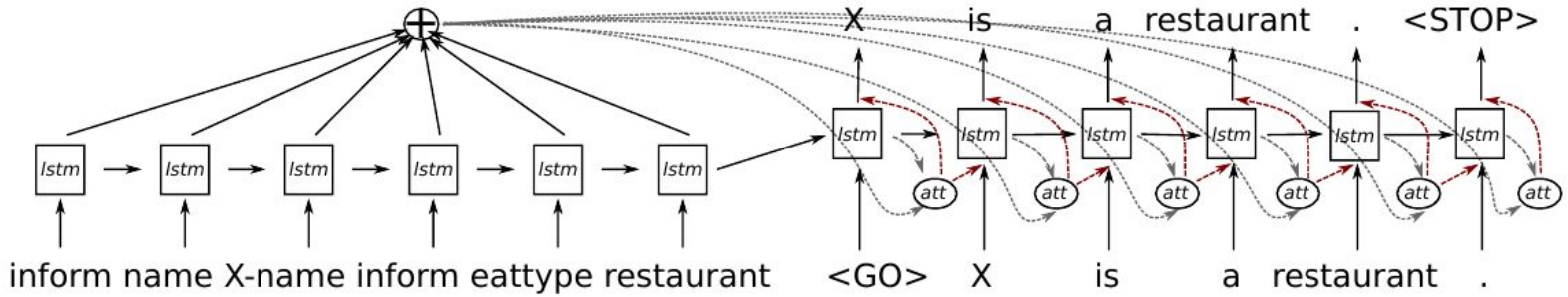


MBR 😊

RNN-based Approaches

- first neural approaches: ~2015
- TGen: standard LSTM with attention
  - input: triples <intent, slot, value>, output: delexicalized text
  - beam search & reranking

(Dušek & Jurčíček, 2016)
https://aclweb.org/anthology/P16-2008



- RNNLM: special LSTM gate cells to control slot mentions
- mitigating the lack of training data for specific entities: delexicalization / copy mechanism

(Wen et al, 2015; 2016)
http://aclweb.org/anthology/D15-1199
http://arxiv.org/abs/1603.01232

(See et al., 2017)
http://arxiv.org/abs/1704.04368

Finetuning PLMs

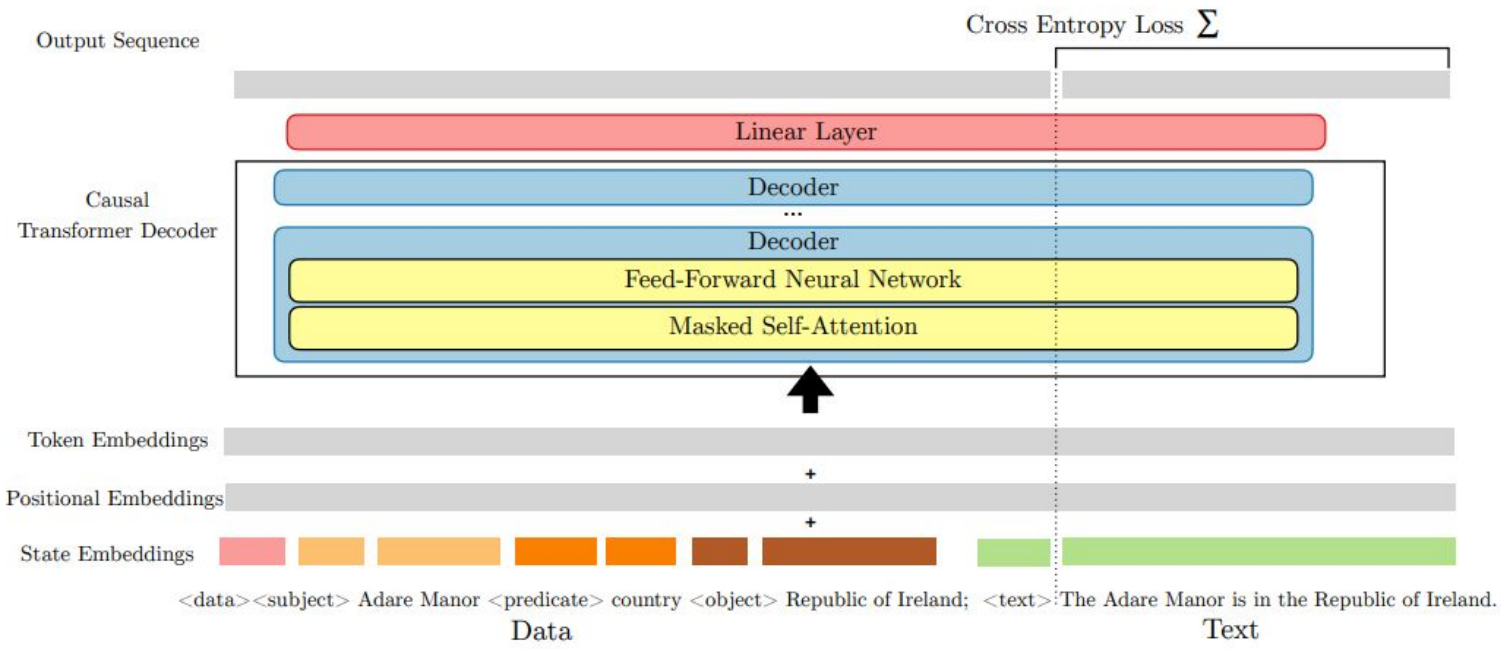
- finetuning Transformer-based PLMs (decoder-only / enc-dec)
- input: structured meaning representation, output: text
  - similar to training RNNs, just starting from pretrained checkpoints
- more fluent than RNNs, implicit copying, can use multilingual models

(Kale & Rastogi, 2020)
https://www.aclweb.org/anthology/2020.inlg-1.14

(Kasner & Dušek, 2020)
https://aclanthology.org/2020.webnlg-1.20/

(Liu et al., 2020)
http://arxiv.org/abs/2001.08210

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



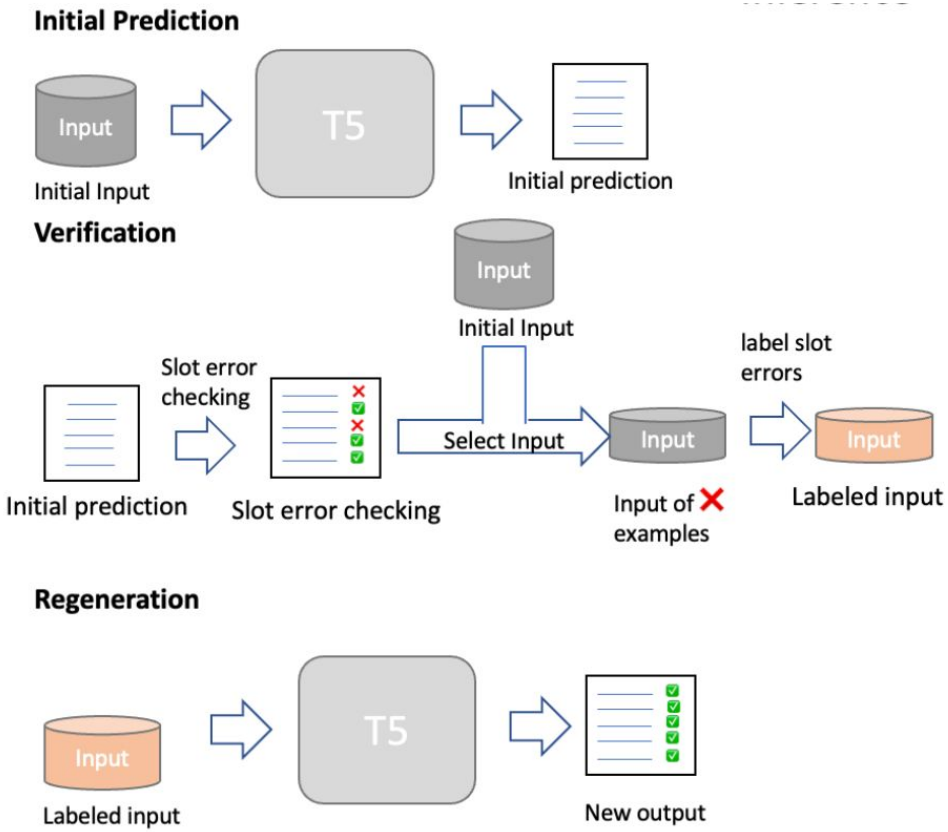
Finetuning PLMs + Reranking

- goal: improving semantic accuracy
- classifying errors in model outputs with a classifier
  - e.g., accurate / omission / repetition / hallucination / value error
- **reranking:** selecting the output with the fewest errors

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

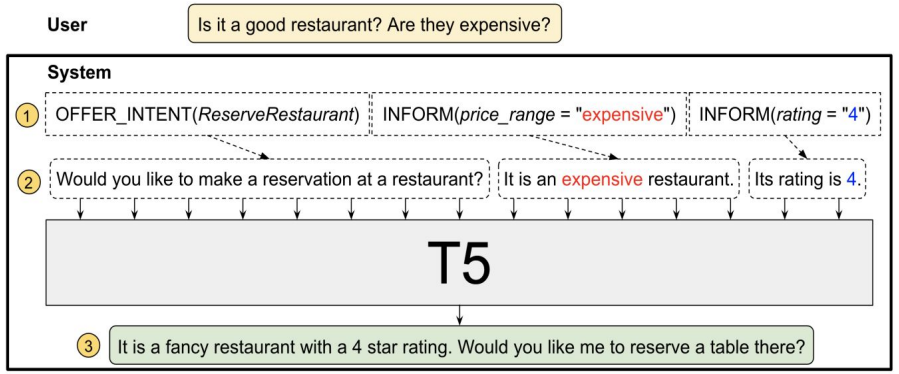
- or regenerating the output (with error labels provided as an extra input)

(Ren and Liu, 2023)
http://arxiv.org/abs/2306.15933

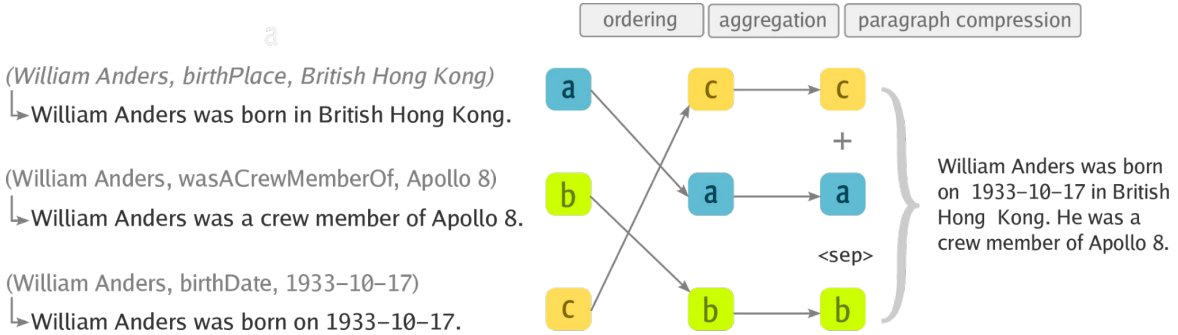


Finetuning PLMs + Templates

- goal: improving semantic accuracy + controllability
- concatenate simple templates and then use PLMs to make the text more fluent
- combines advantages of templates (controllability) and neural LMs (fluency)
- needs less data & generalizes to new domains



(Kale & Rastogi, 2020)
https://www.aclweb.org/anthology/2020.emnlp-main.527



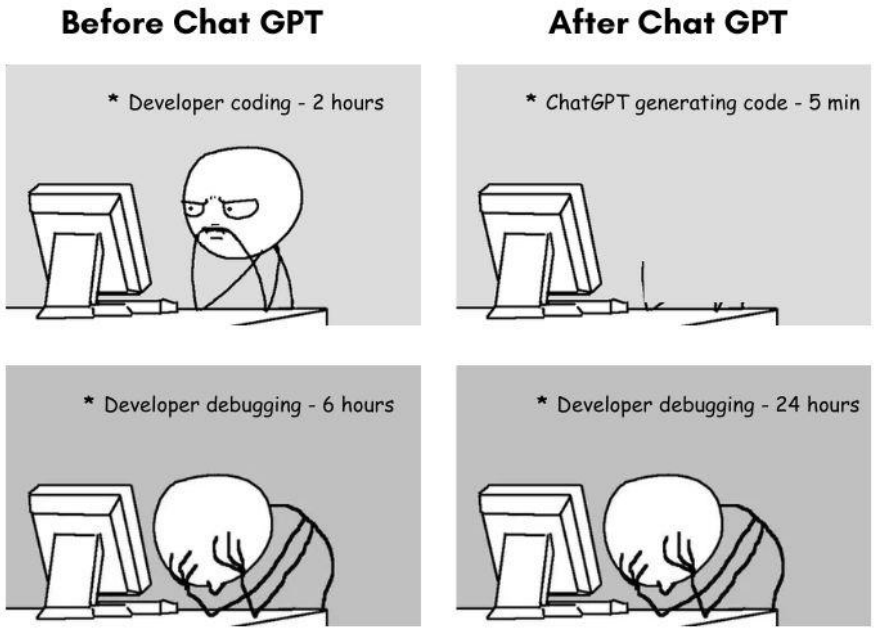
(Kasner & Dušek, 2022)
https://aclanthology.org/2022.acl-long.271/

(PLMs can be also replaced with prompted GPT-3)

(Xiang et al., 2022)
http://arxiv.org/abs/2210.04325

NLG with LLMs

- **prompting** instead of task-specific finetuning (see Lecture 4)
- for NLG, prompting competitive to finetuning, but different kinds of problems:
  - variability in responses ("Here is the answer: (...)", "As an AI language model (...)"
  - prompt sensitivity
  - semantic errors
- many issues with closed models: replicability, cost, data contamination
- more details in our new course NPFL140 (Large Language Models) in the summer term



https://www.boredpanda.com/chatgpt-memes/

(Yuan and Färber, 2023)
http://arxiv.org/abs/2307.14712

(Axelsson and Skantze, 2023)
http://arxiv.org/abs/2307.07312

NLG with LLMs

- fluency much better than PLMs → evaluation needs to focus on **semantic errors**
- data contamination is a serious issue: need to evaluate on novel / non-public data
- can work with standard representations (JSON, CSV, ...) without finetuning
- open LLMs → better replicability (although not perfect transparency
- for NLG in dialog, overgenerate-and-rerank still helps

(Ramirez et al., 2023)
https://aclanthology.org/2023.sigdial-1.32

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}:
"
```

llama2

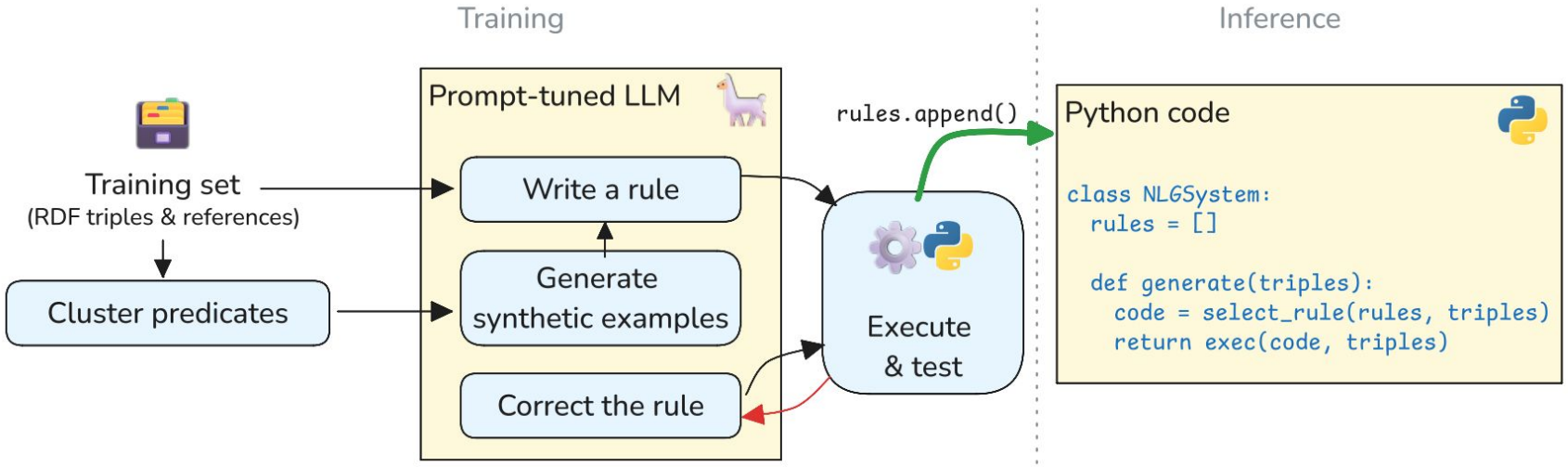
The weather in Holt, England, will be mostly cloudy with scattered clouds for the next five days. The temperature will range from 4.2 to 7.4 degrees Celsius, with a high of 6.1 degrees Celsius on January 7th. The wind speed will be around 2.5 to 4.6 meters per second, with gusts of up to 9.32 meters per second on January 4th. There will be light rain on January 4th, with a total of 0.12 millimeters of precipitation. The humidity will be around 77% to 97%.

(Kasner and Dušek, 2024)
https://aclanthology.org/2024.inlg-main.48

Rule-based NLG system with LLMs

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

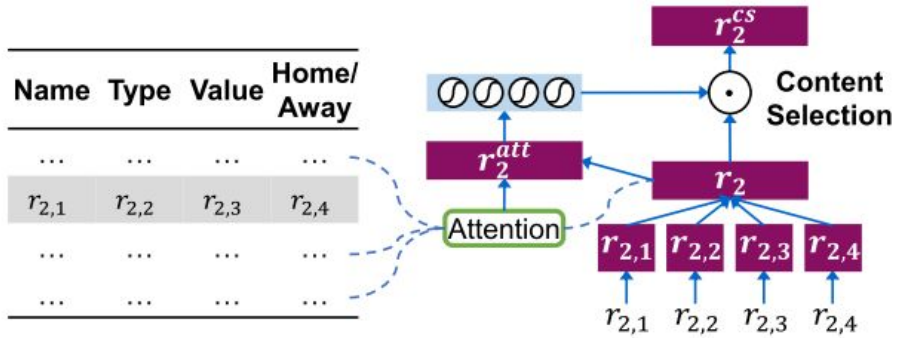
- instead of generating the output directly, we can use **LLMs to generate the rules**
  - e.g. Python string templates
- fewer hallucinations, CPU-only inference
- only limited generalization to unseen data



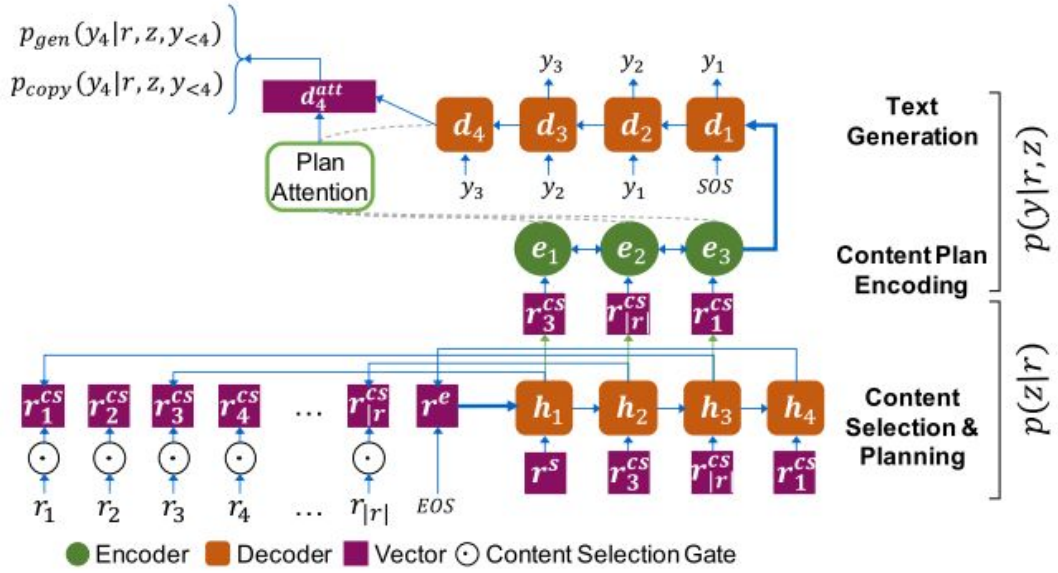
Content Planning: Content Selection

- **explicit content selection**
- usually done by DM in dialogue systems
- needed for complex inputs, e.g. sports report generation
  - records (team / entity / type / value) → summary
  - content selection: pointer network
- still largely unsolved problem w.r.t. semantic accuracy

(Thomson & Reiter, 2022)
http://arxiv.org/abs/2108.05644



(Puduppully et al., 2019)
http://arxiv.org/abs/1809.00582



Content Planning: Content Selection

Example of NLG with content planning

source statistics (excerpt)

TEAM	WIN	LOSS	PTS	FG_PCT	RB	AST	...
Pacers	4	6	99	42	40	17	...
Celtics	5	4	105	44	47	22	...

PLAYER	H/V	AST	RB	PTS	FG	CITY	...
Jeff Teague	H	4	3	20	4	Indiana	...
Miles Turner	H	1	8	17	6	Indiana	...
Isaiah Thomas	V	5	0	23	4	Boston	...
Kelly Olynyk	V	4	6	16	6	Boston	...
Amir Johnson	V	3	9	14	4	Boston	...
...	...	...	...	...	...	...	...

PTS: points, FT_PCT: free throw percentage, RB: rebounds, AST: assists, H/V: home or visiting, FG: field goals, CITY: player team city.

content plan (for the 1st sentence)

Value	Entity	Type	H/V
Boston	Celtics	TEAM-CITY	V
Celtics	Celtics	TEAM-NAME	V
105	Celtics	TEAM-PTS	V
Indiana	Pacers	TEAM-CITY	H
Pacers	Pacers	TEAM-NAME	H
99	Pacers	TEAM-PTS	H
42	Pacers	TEAM-FG_PCT	H
22	Pacers	TEAM-FG3_PCT	H
5	Celtics	TEAM-WIN	V
4	Celtics	TEAM-LOSS	V
Isaiah	Isaiah_Thomas	FIRST_NAME	V
Thomas	Isaiah_Thomas	SECOND_NAME	V
23	Isaiah_Thomas	PTS	V
5	Isaiah_Thomas	AST	V
4	Isaiah_Thomas	FGM	V
13	Isaiah_Thomas	FGA	V
Kelly	Kelly_Olynyk	FIRST_NAME	V
Olynyk	Kelly_Olynyk	SECOND_NAME	V
16	Kelly_Olynyk	PTS	V
6	Kelly_Olynyk	REB	V
4	Kelly_Olynyk	AST	V
...	...	...	...

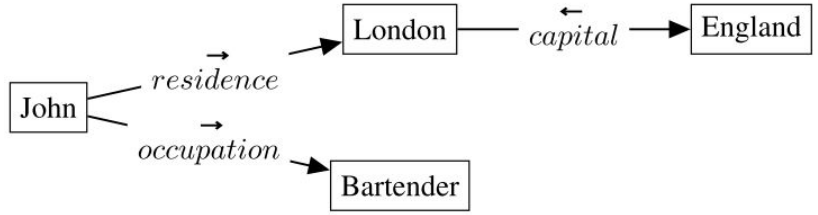
target text

The **Boston Celtics** defeated the host **Indiana Pacers 105-99** at Bankers Life Fieldhouse on Saturday. In a battle between two injury-riddled teams, the Celtics were able to prevail with a much needed road victory. The key was shooting and defense, as the **Celtics** outshot the **Pacers** from the field, from three-point range and from the free-throw line. Boston also held Indiana to **42 percent** from the field and **22 percent** from long distance. The Celtics also won the rebounding and assisting differentials, while tying the Pacers in turnovers. There were 10 ties and 10 lead changes, as this game went down to the final seconds. Boston (**5–4**) has had to deal with a gluttony of injuries, but they had the fortunate task of playing a team just as injured here. **Isaiah** Thomas led the team in scoring, totaling **23 points and five assists on 4–of–13** shooting. He got most of those points by going 14–of–15 from the free-throw line. **Kelly Olynyk** got a rare start and finished second on the team with his **16 points, six rebounds and four assists**.

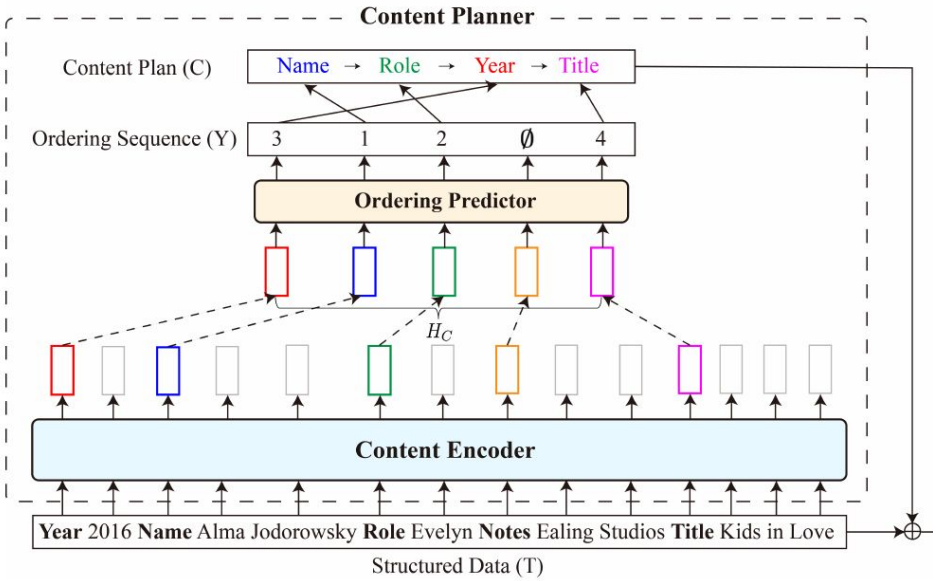
(Puduppully et al., 2019)
http://arxiv.org/abs/1809.00582

Content Planning: Ordering & Aggregation

- **ordering the facts + aggregating them into sentences**
- content already selected at this point
- can help the generator not to miss any facts
- for graphs with oriented edges:
  - generating all possible content plans using DFS (possibly pruning unpromising branches) → re-ranking the plans using a feature-based classifier
- for a set of key-value pairs:
  - using Conditional Random Field (CRF) for finding the optimal plan



(Moryossef et al., 2019a,b)
http://arxiv.org/abs/1904.03396
https://arxiv.org/pdf/1909.09986.pdf



(Su et al., 2020)
http://arxiv.org/abs/2108.13740

Realizing from Trees

(Balakrishnan et al., 2019) http://arxiv.org/abs/1906.07220
(Rao et al., 2019) https://www.aclweb.org/anthology/W19-8611/
(Li et al., 2021) https://aclanthology.org/2021.inlg-1.10

- **NLG with tree-shaped inputs**
- simple case: discourse relations (discourse connectives, sentence splits) between individual fields
  - much flatter than usual syntactic trees
- improvements to account for the input structure:
  - constrained beam search decoding, tree-LSTM, self-training on synthetic data

Reference 1	JJ's Pub is not family friendly, but has a high customer rating of 5 out of 5. It is a restaurant near the Crowne Plaza Hotel.
Reference 2	JJ's Pub is not a family friendly restaurant. It has a high customer rating of 5 out of 5. You can find it near the Crowne Plaza Hotel.
E2E MR	name[JJ's Pub] rating[5 out of 5] familyFriendly[no] eatType[restaurant] near[Crowne Plaza Hotel]
Our MR for Reference 1	**CONTRAST** [**INFORM** [name[JJ's Pub] familyFriendly[no]] **INFORM** [rating[5 out of 5]]] **INFORM** [eatType[restaurant] near[Crowne Plaza Hotel]]

Query	**Context**	**MR**	**Response**
When will it snow next?	Reference date: 29th September 2018	[CONTRAST [INFORM_1 [LOCATION [CITY Parker]] [CONDITION_NOT snow] [DATE_TIME [DAY 29] [MONTH September] [YEAR 2018]]] [INFORM_2 [DATE_TIME [DAY 29] [MONTH September] [YEAR 2018]] [LOCATION [CITY Parker]] [CONDITION heavy rain showers] [CLOUD_COVERAGE partly cloudy]]]	Parker is not expecting any snow, but today there's a very likely chance of heavy rain showers, and it'll be partly cloudy

Annotated Response

[CONTRAST [INFORM_1 [LOCATION [CITY Parker]] is not expecting any [CONDITION_NOT snow]], but [INFORM_2 [DATE_TIME [COLLOQUIAL today]] there's a [PRECIP_CHANCE_SUMMARY very likely chance] of [CONDITION heavy rain showers] and it'll be [CLOUD_COVERAGE partly cloudy]]]

Data Noise & Cleaning

- NLG errors are often caused by **data errors**
  - ungrounded facts (← hallucinating)
  - missing facts (← forgetting)
  - domain mismatch
  - noise (e.g. source instead of target)
    - just 5% untranslated stuff kills an NMT system
- easy-to-get data are noisy
  - web scraping – lot of noise, typically not fit for purpose
  - crowdsourcing – workers forget/don't care
- **cleaning** improves situation a lot
  - can be done semi-automatically up to a point

(Khayrallah & Koehn, 2018)
https://www.aclweb.org/anthology/W18-2709

Original MR and an accurate reference

MR name[Cotto], eatType[coffee shop], food[English], priceRange[less than £20], customer_rating[low], area[riverside], near[The Portland Arms]

Reference At the riverside near The Portland Arms, Cotto is a coffee shop that serves English food at less than £20 and has low customer rating.

Example corrections

Reference: Cotto is a coffee shop that serves English food in the city centre. They are located near the Portland Arms and are low rated.
Correction: removed price range; changed area
Reference: Cotto is a cheap coffee shop with one-star located near The Portland Arms.
Correction: removed area

A faulty correction

Reference: Located near The Portland Arms in riverside, the Cotto coffee shop serves English food with *a price range of $20* and a low customer rating.
Correction: incorrectly(!) removed price range
– our script's slot patterns are not perfect

(Dušek et al., 2019)
https://arxiv.org/abs/1911.03905

(Wang, 2019)
https://www.aclweb.org/anthology/W19-8639/

Summary

- **NLG**: system action → system response
- **templates** work pretty well
- **seq2seq generation** with finetuned PLMs
  - best among data-driven
  - problems – hallucination, not enough diversity, needs lots of data
- **prompting-based** approaches with LLMs
  - less effort than finetuning
  - problems – hallucination, controllability, prompt sensitivity, model access
- mitigating problems: re-ranking, modularization, data cleaning

Thanks

Contact us:

https://ufaldsg.slack.com/
{kasner,odusek}@ufal.mff.cuni.cz
Skype/Meet/Zoom/Troja (by agreement)

Get these slides here:

http://ufal.cz/npfl099

Labs in 10 minutes
Assignment 4

Next week: Dialogue Management (part 2)

References/Inspiration/Further:

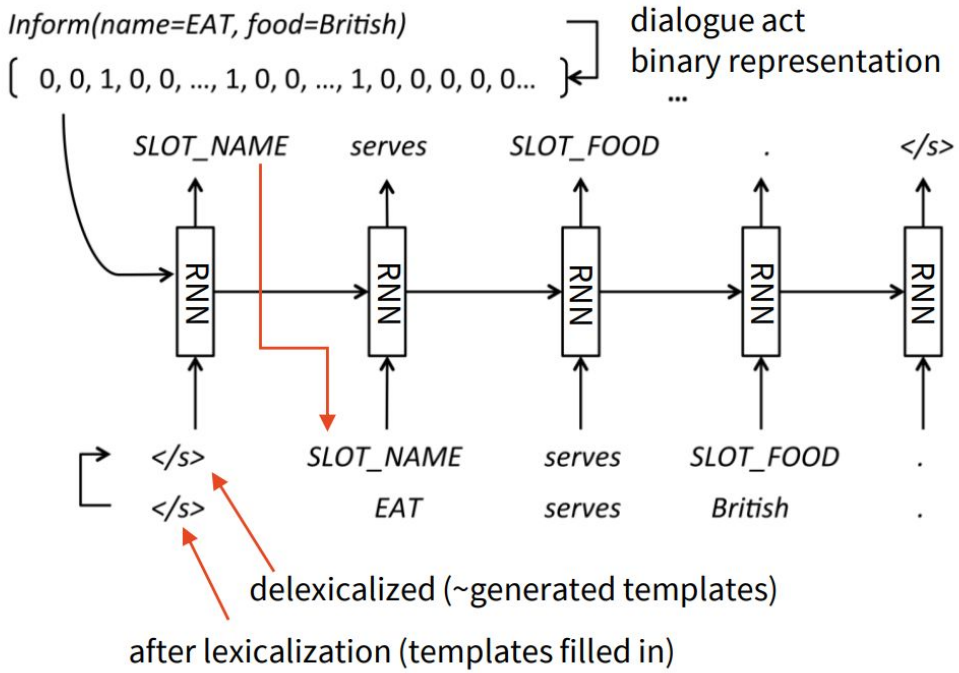
- Reiter (2024). Natural Language Generation. https://link.springer.com/book/10.1007/978-3-031-68582-8 (paid-only access for now)
- Zdeněk's PhD thesis (2024): https://dspace.cuni.cz/handle/20.500.11956/193018
- Sharma et al. (2022). Innovations in Neural Data-to-text Generation. https://arxiv.org/pdf/2207.12571.pdf
- Ondřej's PhD thesis (2017), especially Chapter 2: http://ufal.mff.cuni.cz/~odusek/2017/docs/thesis.print.pdf
- Gatt & Krahmer (2017): Survey of the State of the Art in Natural Language Generation: Core tasks, applications and evaluation http://arxiv.org/abs/1703.09902

Icons from https://www.flaticon.com/

RNN-based Approaches: RNNLM

(Wen et al, 2015; 2016)
http://aclweb.org/anthology/D15-1199
http://arxiv.org/abs/1603.01232

- using enhanced LSTM cells (SC-LSTM)
  - special gate to control slot mentions
- autoregressive generation
- conditioned on DA represented as binary vector
  - generating **delexicalized texts**
- domain adaptation
  - replacing delexicalized slots
  - very related domains only



Few-shot NLG with Pretrained LMs

- **learning from very few (less than ~200) training examples**
- GPT-2 with a copy mechanism
  - LM fine-tuned, forced to copy inputs
  - additional loss term for copying
- retrieving "prototypes" guiding the generator
  - prototype: most similar exemplar according to BERT cosine similarity
  - prototype concatenated with the input
- few-shot prompting
  - prepending a few (~3) input-output examples as a context
  - generating the output with GPT-2 XL
  - no finetuning

(Chen et al., 2020)
https://www.aclweb.org/anthology/2020.acl-main.18/

(Su et al., 2021)
http://arxiv.org/abs/2108.12516

(Keymanesh et al., 2022)
http://arxiv.org/abs/2205.11505

Few-shot Prompt

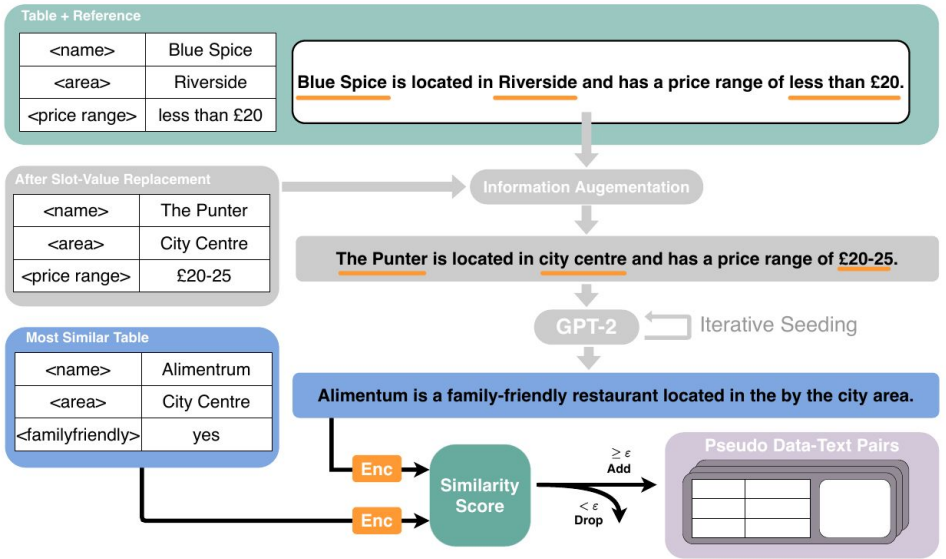
Translate Graph to English:

Graph: <H> Paulo Sousa <R> CLUB <T> ACF Fiorentina
English: Paulo Sousa plays for ACF Fiorentina.
###
Graph: <H> Dave Challinor <R> CLUB <T> Colwyn Bay F.C.
English: Dave Challinor plays for Colwyn Bay F.C.
###
Graph: <H> Alan Martin (footballer) <R> CLUB <T> Hamilton Academical F.C.
English:

Data Augmentation

- **using synthetic data for improving model performance and robustness**
- quite hard for NLG
- where to get the data:
  - extracting (noisy) structured input from unlabeled text
    - keywords, information extraction
  - recombination of existing data inputs
    - model-generated outputs → filtering based on cycle consistency
  - paraphrasing existing data outputs
- how to apply it:
  - task-specific pretraining on synthetic data
  - mixing synthetic data with the training data

(Elder et al., 2020)
https://www.aclweb.org/anthology/2020.acl-main.665
(Montella et al., 2020)
https://arxiv.org/pdf/2012.00571.pdf
(Chang et al., 2021)
http://arxiv.org/abs/2102.03556
(Lee et al. 2021)
http://arxiv.org/abs/2110.06800



Template-based NLG – Examples

Example: Dialogue assistants

Alexa

On the **Intents** detail page, the **Intent Slots** section after the **Sample Utterances** section displays the slots you add. When you highlight a word or phrase in an utterance, you can add a new slot or select an existing slot.

For example, the set of utterances shown earlier now looks like the following example.

```
i am going on a trip on {travelDate}
i want to visit {toCity}
I want to travel from {fromCity} to {toCity} {travelDate}
I'm {travelMode} from {fromCity} to {toCity}
i'm {travelMode} to {toCity} to go {activity}
```

(https://developer.amazon.com/en-US/docs/alexa/custom-skills/create-intents-utterances-and-slots.html)

Mycroft

```
Order some {food}.
Order some {food} from {place}.
Grab some {food}.
Grab some {food} from {place}.
```

Rather than writing out all combinations of possibilities, you can embed them into a single line by writing each possible option inside parentheses with | in between each part. For example, that same intent above could be written as:

```
(Order | Grab) some {food} (from {place} | )
```

(https://mycroft-ai.gitbook.io/docs/mycroft-technologies/padatious)

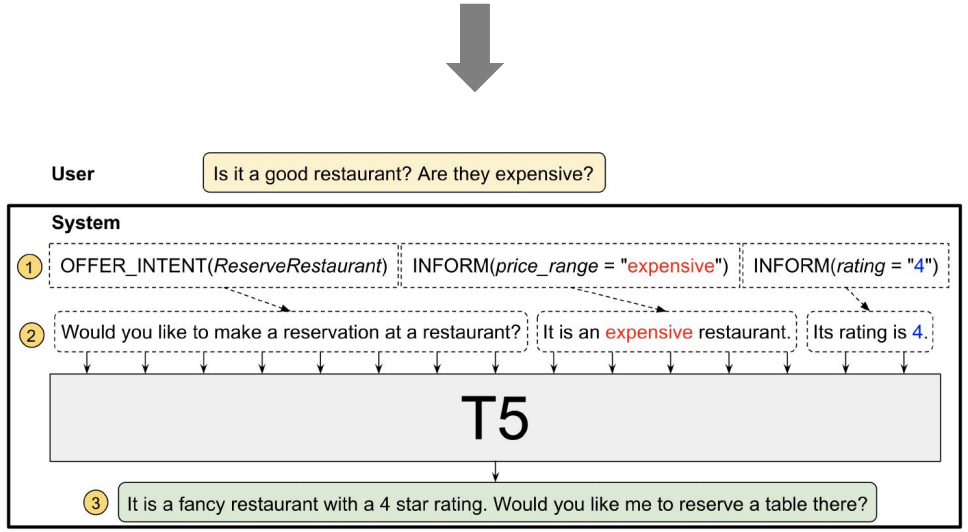
Template-based NLG – Examples

Example: Research systems

```
'iconfirm(to_stop={to_stop})&iconfirm(from_stop={from_stop})':
    "Alright, from {from_stop} to {to_stop},",

'iconfirm(to_stop={to_stop})&iconfirm(arrival_time_rel="{arrival_time_rel}")':
    "Alright, to {to_stop} in {arrival_time_rel},",

'iconfirm(arrival_time="{arrival_time}")':
    "You want to be there at {arrival_time},",

'iconfirm(arrival_time_rel="{arrival_time_rel}")':
    "You want to get there in {arrival_time_rel},",
```

(Alex public transport information rules)
https://github.com/UFAL-DSG/alex

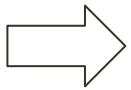
CONFIRM!!date!!@	The date is @.
CONFIRM!!party_size!!@	The reservation is for @ people.
CONFIRM!!restaurant_name!!@	Booking a table at @.
CONFIRM!!time!!@	The reservation is at @.
GOODBYE	Have a good day.
INFORM!!cuisine!!@	They serve @ kind of food.
INFORM!!has_live_music!!False	They do not have live music.
INFORM!!has_live_music!!True	They have live music.

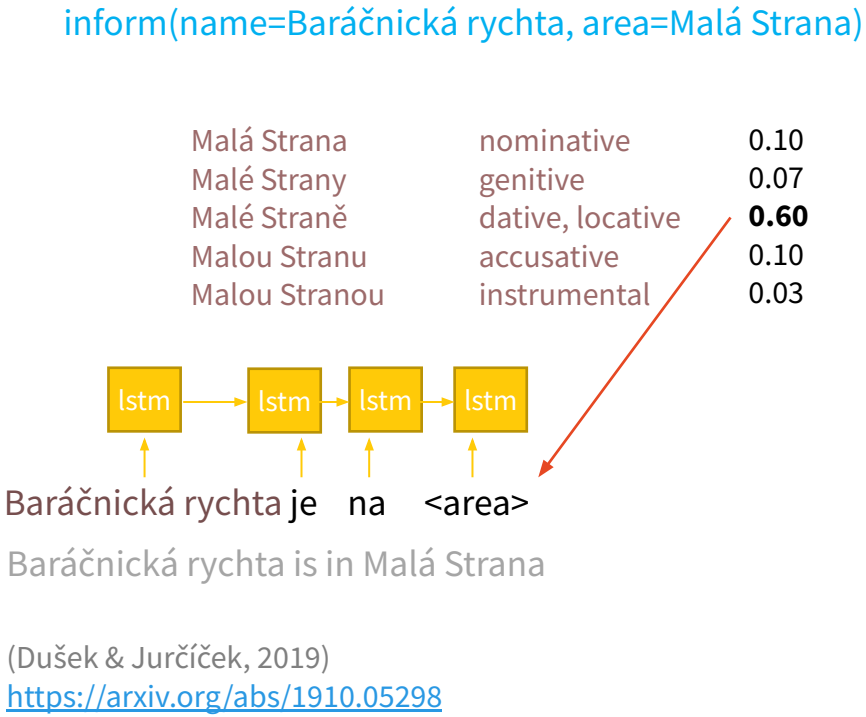


(Kale & Rastogi, 2020)
https://www.aclweb.org/anthology/2020.emnlp-main.527

Delexicalization Alternatives

- **copy mechanism** (see NLU & the next slide)
  - select (or interpolate) between:
    - generating a new token
    - copying a token from input
  - removes the need for pre/postprocessing
- **inflection model**
  - useful for languages with rich morphology (e.g., Czech)
  - a simple LM such as RNN LM
- **pretrained models**
  - the model learns to copy and inflect words implicitly during pretraining
  - works well for high-resource languages



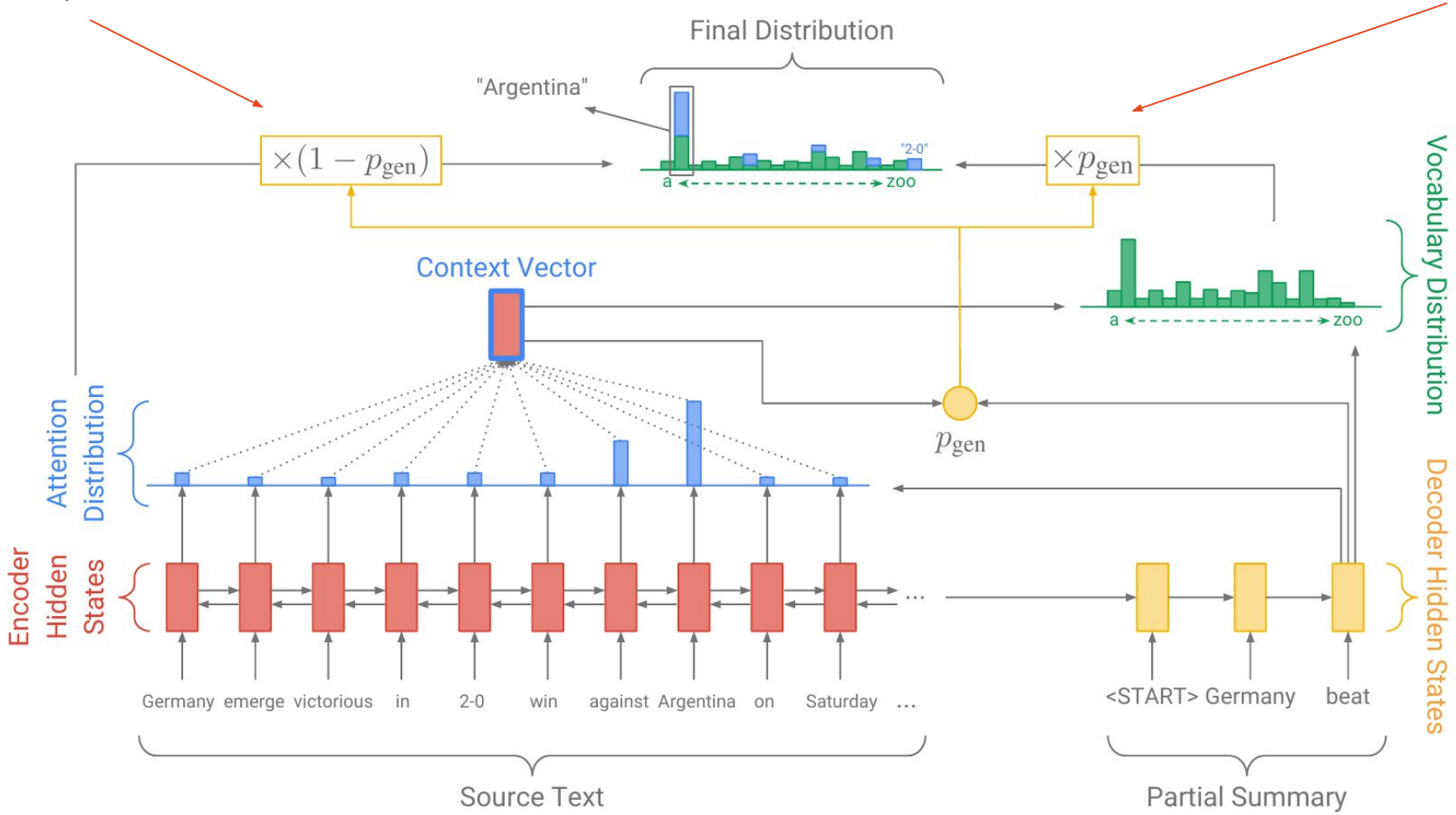


(Dušek & Jurčíček, 2019)
https://arxiv.org/abs/1910.05298

Delexicalization Alternatives – Copy Mechanism

probability of copying a token from the input

probability of generating a new token from the vocabulary



(See et al., 2017) http://arxiv.org/abs/1704.04368