CS310 Natural Language Processing
自然语言处理
Lecture 09 - Natural Language Generation

Instructor: Yang Xu

主讲人：徐炀

xuyang@sustech.edu.cn

Overview

- NLG Taxonomy
- Review of Language Models
- Decoding from NLG models
- Evaluating NLG Systems

What is NLG?

- Natural language Processing (**NLP**) =
  Natural Language Understanding (**NLU**) +
  Natural Language Generation (**NLG**)

- NLG focuses on systems that produce fluent, coherent and useful language output for human consumption

Examples of NLG

- Machine Translation systems
  input: utterances in source languages
  output: translated text in target languages.

- Digital assistant (dialogue) systems
  Older: Siri, Alexa, 小爱同学
  input: dialog history
  output: text that respond / continue the conversation

- Summarization systems
  Writing assistive tools
  input: long documents
  output: shorter pieces of text

GPT-based LLMs are SOTA NLG Systems

- General purpose and capable of many NLG tasks

Can do translation, dialogue, summarization etc. at the same time
-- “An all-in-one toolbox”

Categorization of NLG tasks

- Spectrum of generation tasks, based on the dimension of **open-endedness**

Less Open-ended — More Open-ended

- **Open-ended generation**: the output distribution has high freedom
- **Non-open-ended generation**: the input mostly determines the output generation.

Slide adapted from: https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1244/

Categorization of NLG tasks

Less Open-ended

More Open-ended

Machine translation

Summarization

Source Sentence: 当局已经宣布今天是节假日。

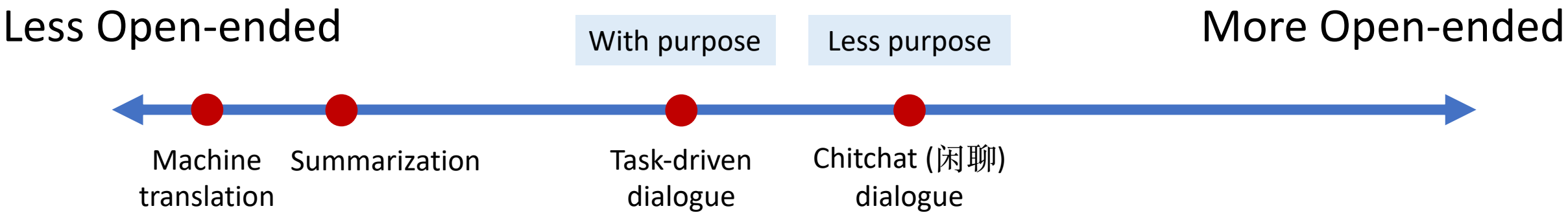
Reference Translation:

1. Authorities have announced a national holiday today.
2. Authorities have announced that today is a national holiday.
3. Today is a national holiday, announced by the authorities.

Output space is not very diverse

or, low entropy

Categorization of NLG tasks



Input: Hey, how are you?

Outputs:

1.Good! You?

2.I just heard an exciting news, do you want to hear it?

3.Thx for asking! Barely surviving my hws.

Output space is getting more diverse

Entropy getting higher

Categorization of NLG tasks

Less Open-ended

More Open-ended

Input: 写一个关于哪吒的剧本杀

Outputs:

“剧本杀名称：《哪吒：神魔之劫》背景设定：在远古的洪荒时代，神魔两界纷争不断。人间界也受到了波及……”

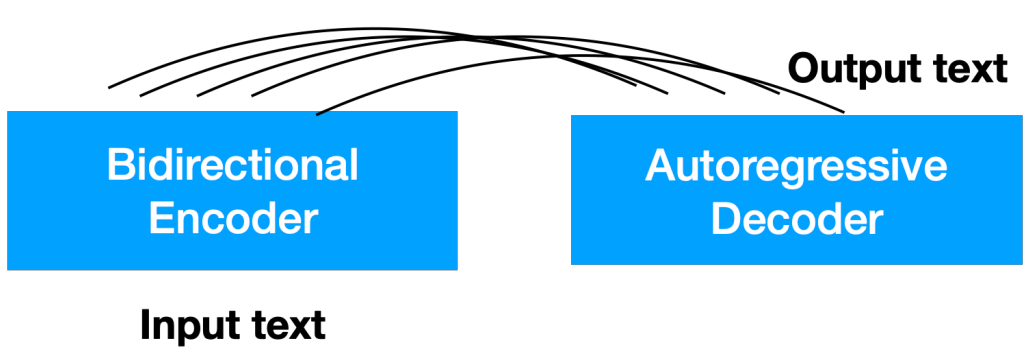
Output space is extremely diverse

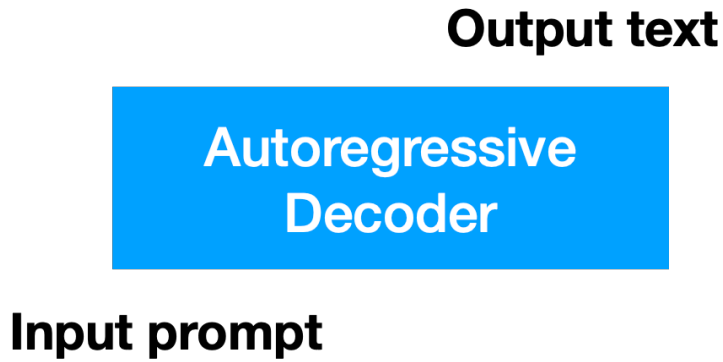
Overview

- NLG Taxonomy
- **Review of Language Models**
- Decoding from NLG models
- Evaluating NLG Systems

Review: Basics of Decoder

- Non-open-ended tasks (e.g., machine translation) => encoder-decoder
- Open-ended tasks (e.g., story generation) => decoder only

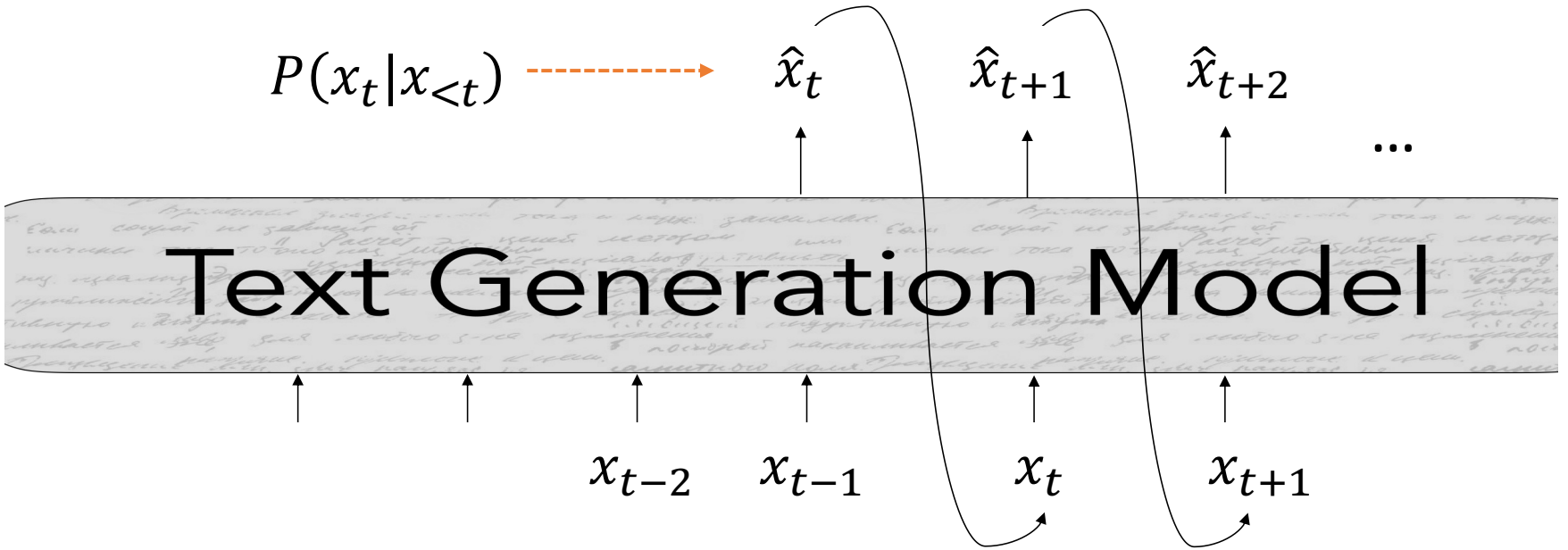




Slide adapted from: https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1244/

Review: Basics of Language Models

At each time step t, the model takes a sequence of tokens as input $\{x\}_{<t}$ and outputs a new token $\hat{x}_t$, by sampling a word from the prob. distr. $P(x_t|x_{<t})$



Slide adapted from: https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1244/

Decoding from Language Model

- Decoding algorithm largely determines how tokens are selected

$$\hat{x}_t = g\big(P(x_t|x_{<t})\big) \qquad g(\cdot) \text{ is the decoding algorithm}$$

- An "obvious" choice for $g(\cdot)$ is to greedily select the token of highest probability
- .. not necessarily the optimal way

Overview

- Motivation: What is NLG?
- Review of Language Models
- **Decoding from NLG models**
  - **Greedy/beam search**
  - **Sampling: top-k, top-p, temperature, re-ranking**
- Evaluating NLG Systems

Decoding

- At each time step t, the LM computes a vector of scores for each token in the vocabulary, $S \in \mathbb{R}^V$

$$S = f(\{y_{<t}\})$$

$f(\cdot)$ is the language model

- Then, compute a probability distribution P over these scores with a softmax function:

$$P(y_t = w|\{y_{<t}\}) = \frac{\exp(S_w)}{\sum_{w' \in V} \exp(S_{w'})}$$

- The decoding algorithm defines a function to select a token from this distribution

$$\hat{y}_t = g\big(P(y_t = w|\{y_{<t}\})\big)$$

$g(\cdot)$ is the decoding algorithm

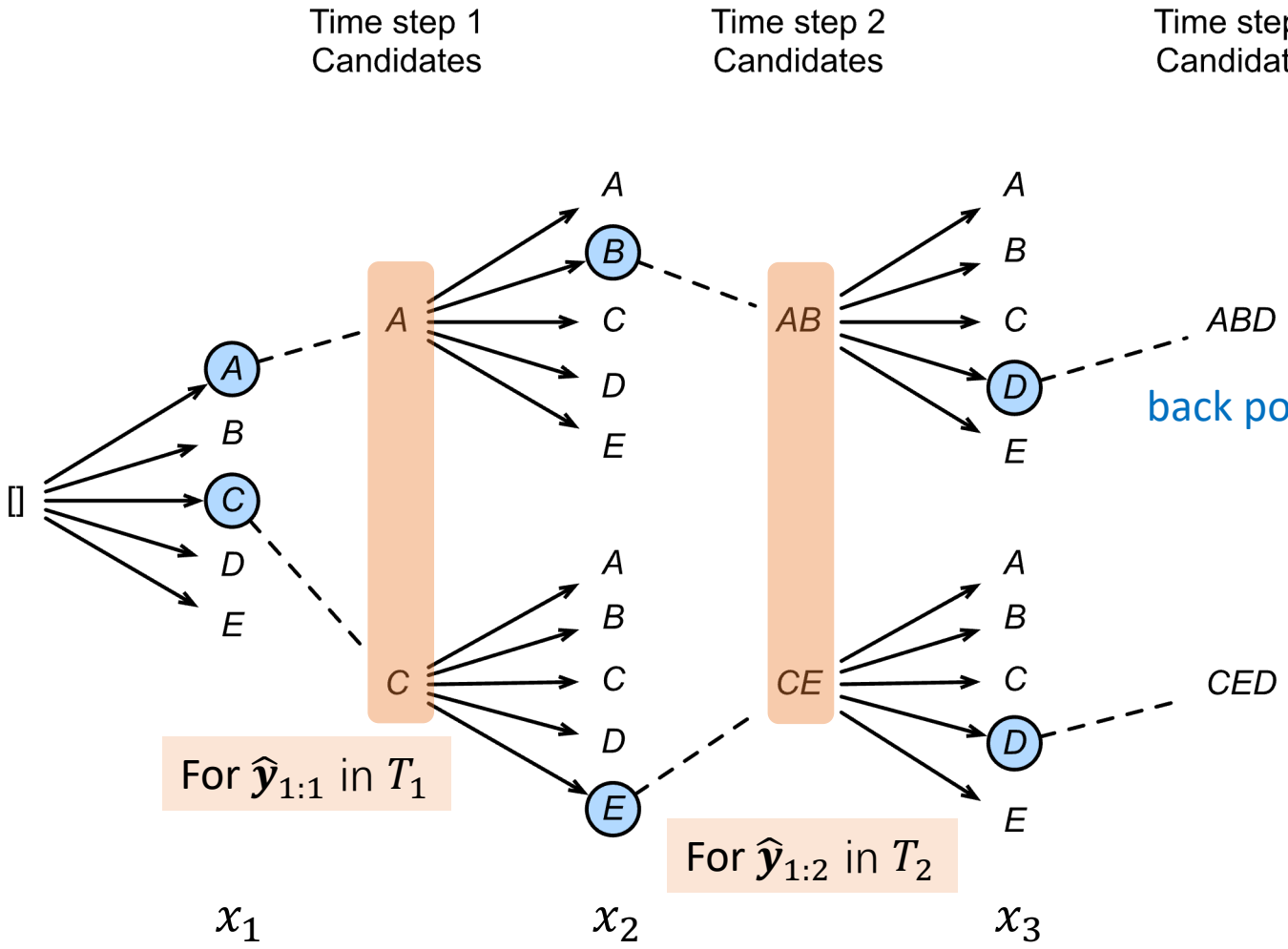
Common ways to find the most likely string

- **Greedy decoding**: Selects the highest probability token in $P(y_t|y_{<t})$

$$\hat{y} = \arg\max_{w \in V} P(y_t = w|y_{<t})$$

- **Beam search**: aims to find strings that maximize the log-prob, but with wider exploration of candidates

Beam Search



Beam width $k = 2$, sequence length $n = 3$

$T_0 = \emptyset$

$T_1 = \{A, C\} \Rightarrow$ top 2 candidates for step 1

back pointers

$T_2 = \{B, E\} \Rightarrow$ top 2 candidates for step 2

$T_3 = \{D, D\} \Rightarrow$ top 2 candidates for step 3

The best label sequences are among:

$\{A \rightarrow B \rightarrow D, C \rightarrow E \rightarrow D\}$

pick one using some standards

Example from: https://d2l.ai/chapter_recurrent-modern/beam-search.html

Problem: The most likely generations are repetitive

Context:

In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

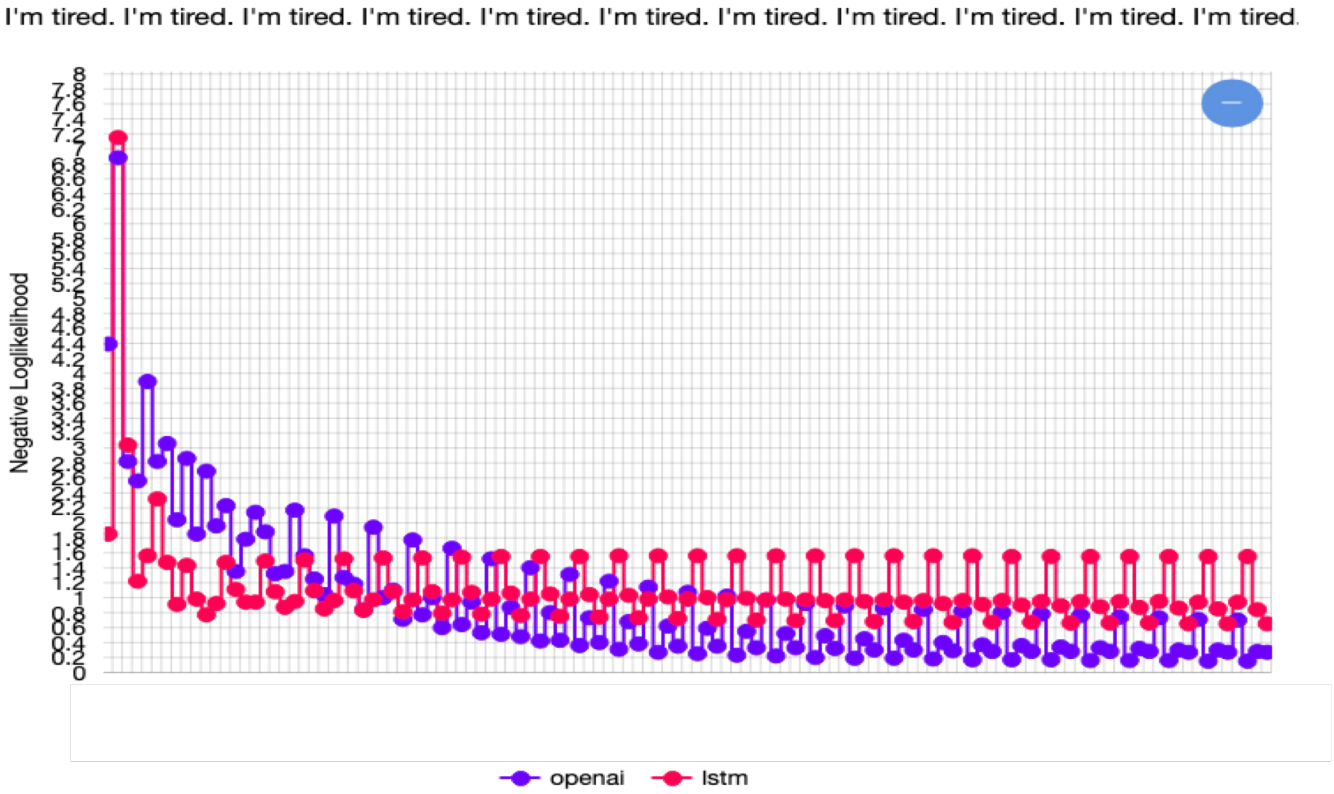
Generated continuation:

using beam search on GPT-2

The study, published in the Proceedings of the National Academy of Sciences of the United States of America (PNAS), was conducted by researchers from **the Universidad Nacional Autónoma de México (UNAM)** and the Universidad Nacional Autónoma de México (UNAM/Universidad Nacional Autónoma de México/ Universidad Nacional Autónoma de México/ Universidad Nacional Autónoma de México/ Universidad Nacional Autónoma de México...

Examples from Holtzman et. al., 2020

Scale Does Not Solve Repetition!

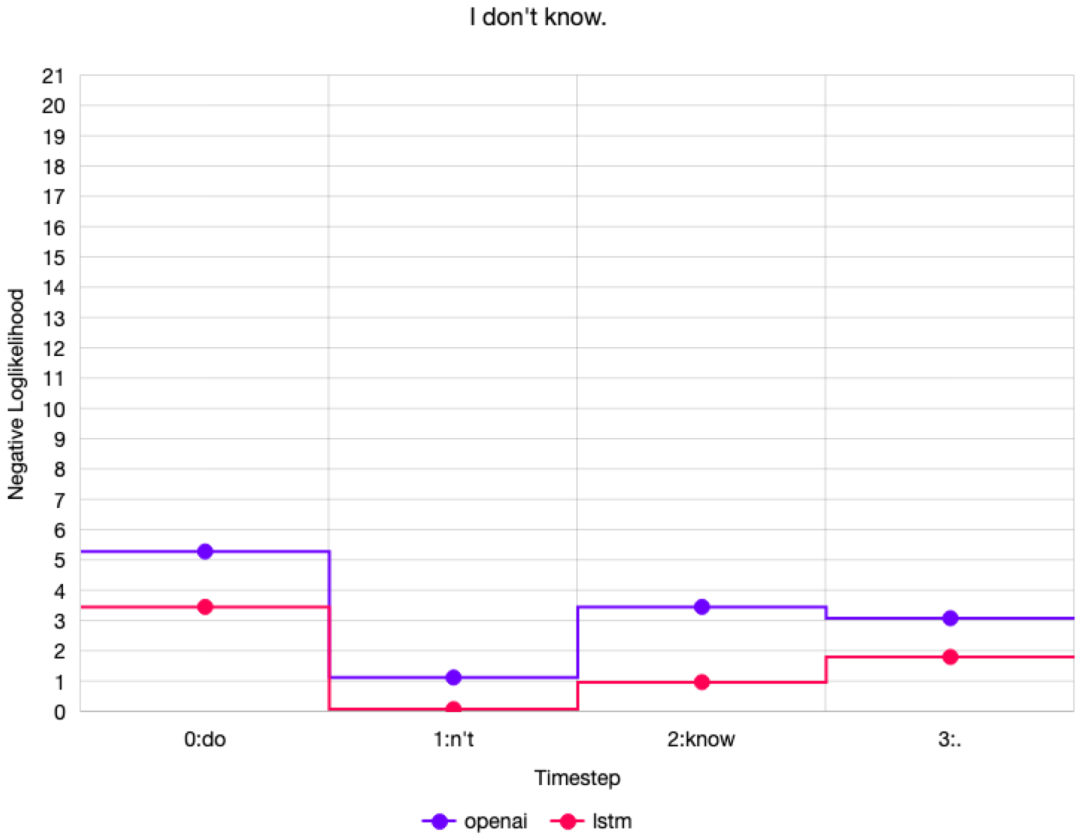


Even a 175 billion parameter LM still repeats when decoding for the most likely string

Figure from Holtzman et. al., 2020

Why does repetition happen?

- Repetition has a self-amplification effect!



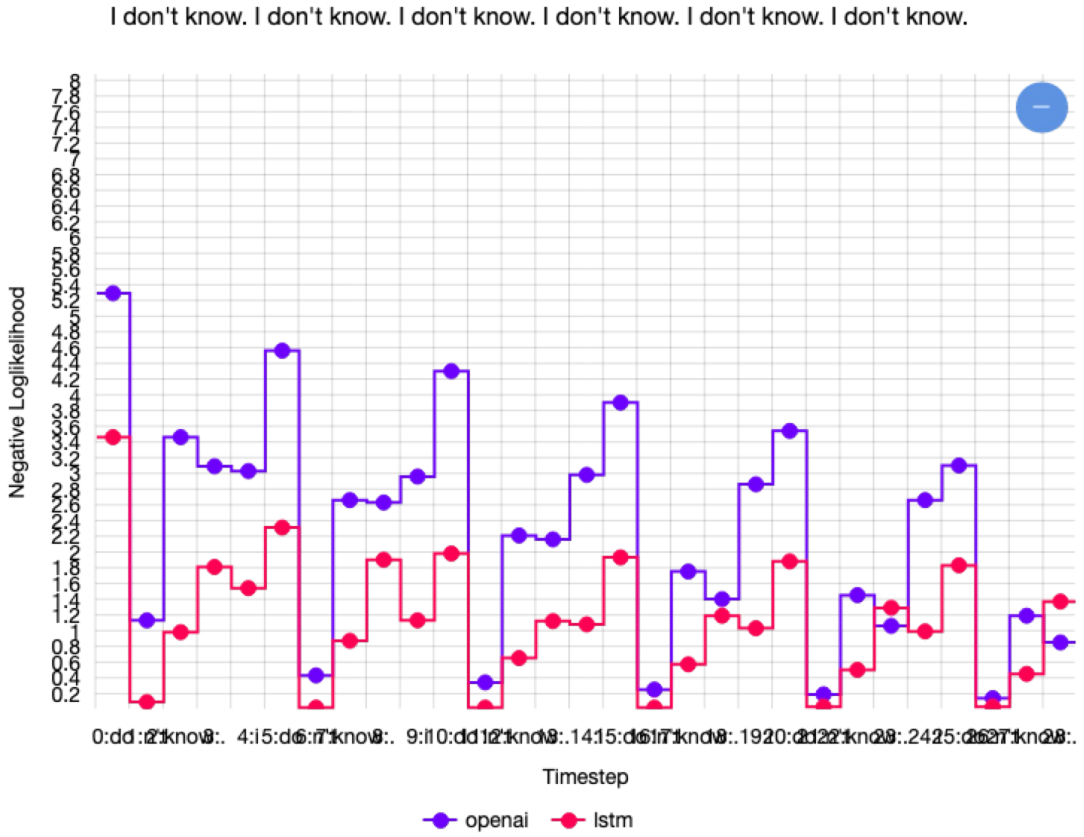
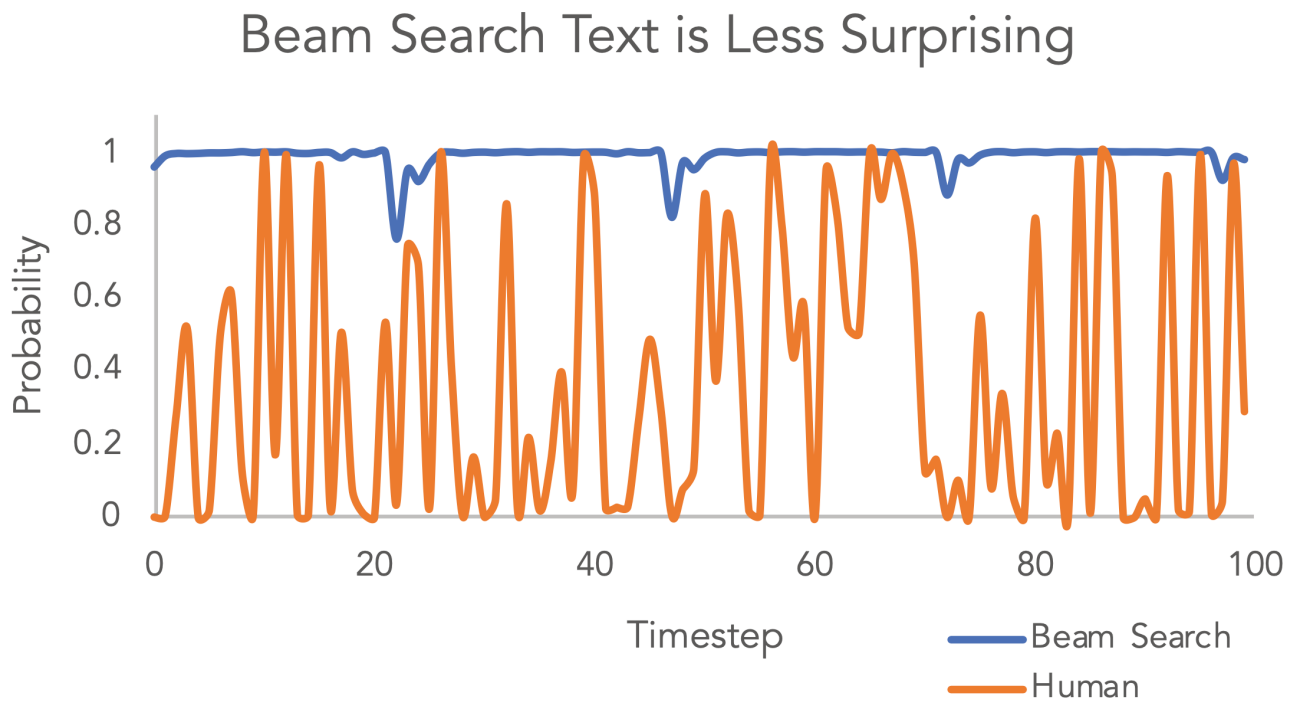


Figure from Holtzman et. al., 2020

Is the most likely string really **reasonable**?



It fails to match the uncertainty distribution for human written text

Human languages are not always predictable …

Figure from Holtzman et. al., 2020

How to reduce repetition?

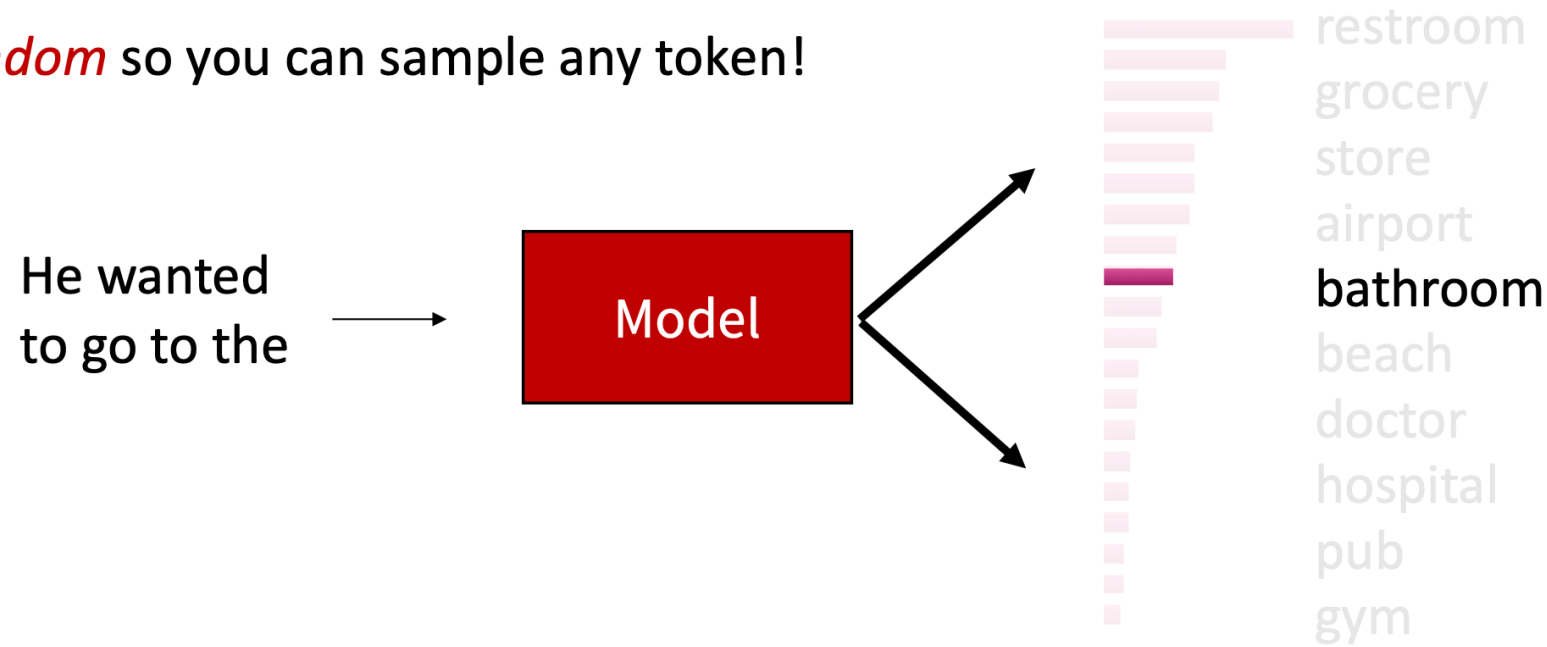
- **Simple option:** Use heuristic: Don't repeat n-grams
- **More complex solutions:**
- Use a different training objective:
  - Unlikelihood objective (Welleck et al., 2020):
    Penalizes generation of already-seen tokens
  - Coverage loss (See et al., 2017):
    Prevents attention mechanism from attending to the same words
- Use a different decoding objective:
  - Contrastive decoding (Li et al, 2022) searches for strings s that maximize:
    $P_{\text{LargeLM}}(s) - P_{\text{SmallLM}}(s)$
  - Contrastive search (Su et al., 2022) to encourage diversity while maintaining coherence in the generated text

More common solution: sampling

Sample a token from the distribution of tokens

$$\hat{y}_t \sim P(y_t = w \mid \{y\}_{<t})$$

It's *random* so you can sample any token!



Top-k Sampling

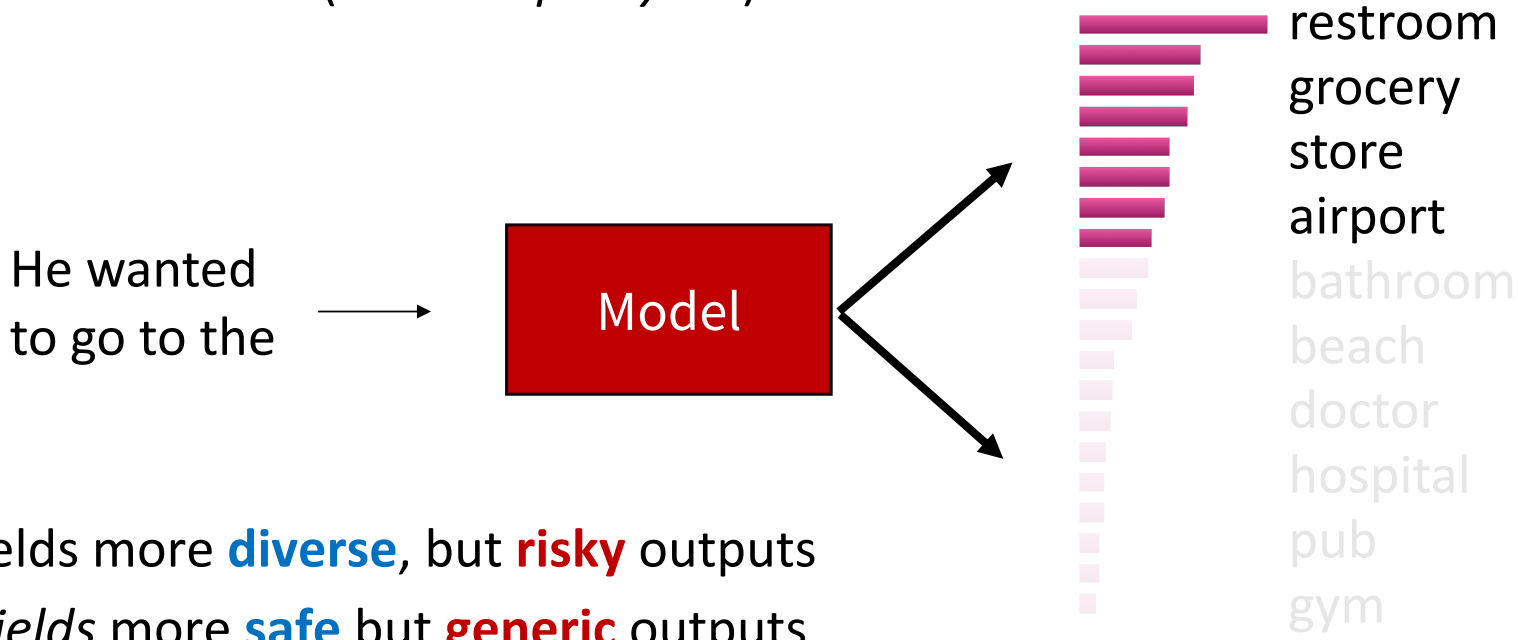
- **Problem of pure sampling**: makes every token in the vocabulary an option
  - Even if most of the probability mass in the distribution is over a limited set of options, the tail of the distribution could be very long and in aggregate have considerable mass (“heavy tailed” distributions)
  - Many tokens are probably *really wrong* in the current context
    - We give them *individually* a tiny chance to be selected.
    - But there are two many of them -- we still give them *as a group* a high chance to be selected.

- **Solution: Top-*k* sampling**
  - Only sample from the top *k* tokens in the probability distribution

Top-*k* Sampling

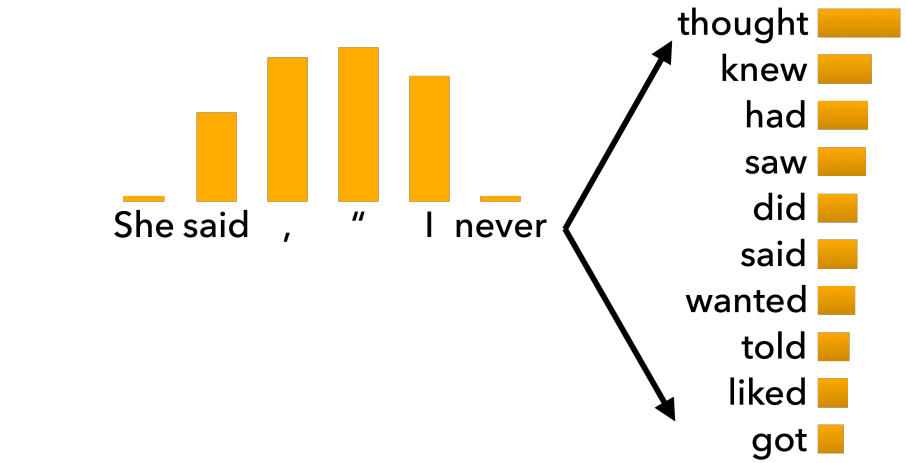
Solution: Top-*k* sampling

- Only sample from the top *k* tokens in the probability distribution
- Common values are *k* = 50 (*but it's up to you!*)

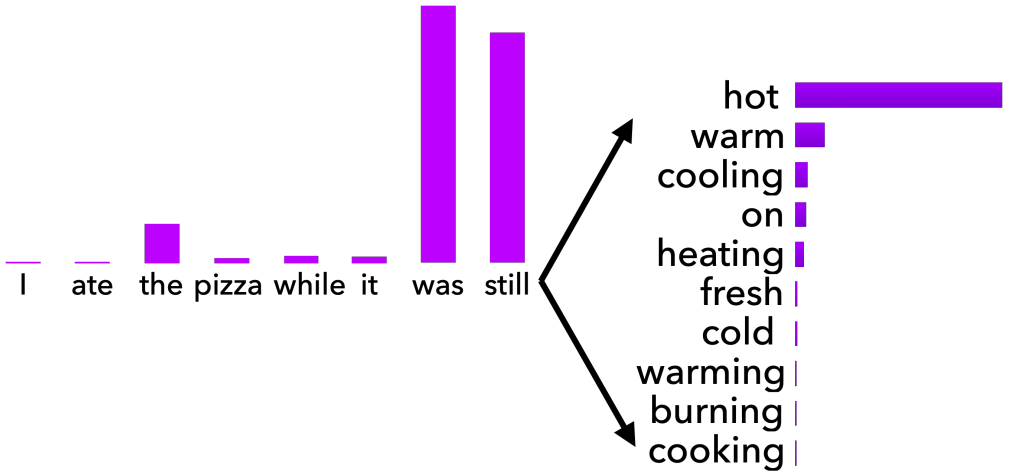


- Increase *k* yields more **diverse**, but **risky** outputs
- Decrease *k yields* more **safe** but **generic** outputs

Issues of Top-k Sampling



Top-*k* sampling can cut off too ***quickly***!



Top-*k* sampling can also cut off too ***slowly***!

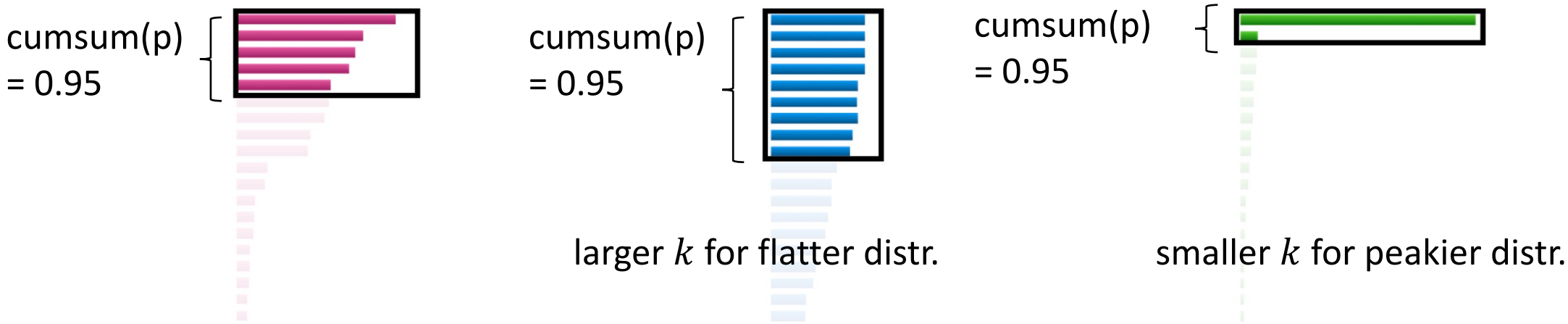
Top-*p* (nucleus) sampling

- Problem: The probability distributions we sample from are dynamic
  - When the distribution P_t is flatter, a small k removes many viable options
  - When the distribution P_t is peakier, a large k allows for too many

- Solution: Top-p sampling
  - Sample from all tokens in the **top p cumulative probability mass** (i.e., where mass is concentrated)
  - Varies k depending on the uniformity of P_t

Top-*p* (nucleus) sampling

- Solution: Top-*p* sampling
  - Sample from all tokens in the top *p* cumulative probability mass (i.e., where mass is concentrated)
  - Varies *k* depending on the uniformity of P_t



Scaling randomness: Temperature

- Add a temperature hyperparameter τ to the softmax to rebalance P_t

Before: $$P(y_t = w|\{y_{<t}\}) = \frac{\exp(S_w)}{\sum_{w' \in V} \exp(S_{w'})}$$

After: $$P(y_t = w|\{y_{<t}\}) = \frac{\exp(S_w/\tau)}{\sum_{w' \in V} \exp(S_{w'}/\tau)}$$

- "High temperature melts matter"
- Raise temperature $\tau > 1$: P_t becomes more uniform, more diverse output, higher entropy;
- Lower temperature $\tau < 1$: P_t becomes more spiky, less diverse output, lower entropy;

Improving Decoding: Re-ranking

- What if I decode a bad sequence from my model?
- **Solution**: Decode a bunch of sequences (.e.g., 10 candidates), and define a score to approximate quality of sequences and **re-rank by this score**
- Simplest option: perplexity
  - Beware that repetitive texts are of low perplexity
- Re-rank based on a variety of properties:
  - Style, discourse, entailment, logical consistency, etc.

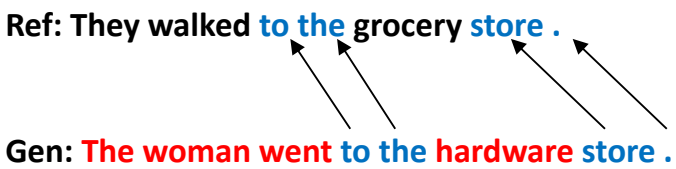
Recap

- Decoding is still a challenging problem in NLG
- Different decoding algorithms can allow us to inject biases that encourage different properties of coherent natural language generation
- Some of the most impactful advances in NLG of the last few years have come from **simple but effective** modifications to decoding algorithms

Overview

- NLG Taxonomy
- Review of Language Models
- Decoding from NLG models
- **Evaluating NLG Systems**

Evaluation Methods for NLG

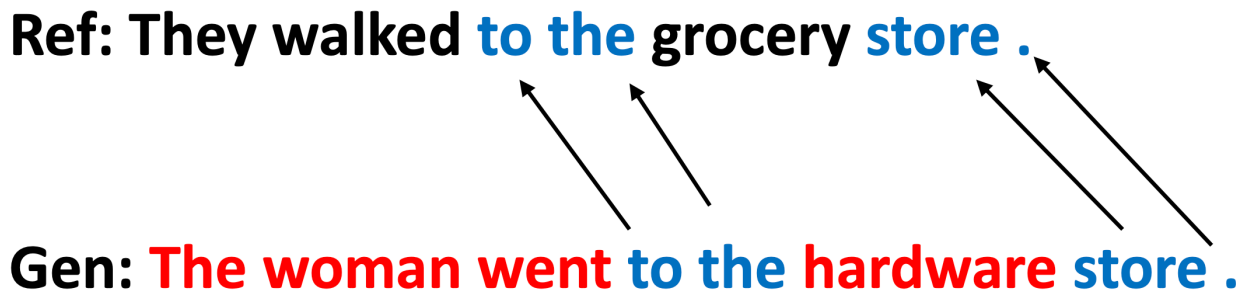


Automatic metrics

Model-based metrics

Human evaluations

Content overlap metrics



- Compute a score that indicates the lexical similarity between generated and gold-standard (human-written) text
- Fast and efficient and widely used
- *N*-gram overlap metrics (e.g., BLEU, ROUGE, METEOR, CIDEr, etc.)

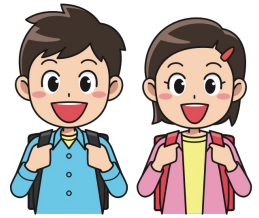
N-gram overlap metrics

- Word overlap–based metrics (BLEU, ROUGE, METEOR, CIDEr, etc.)

- Problem: They're not ideal for machine translation
- Further, they get progressively much worse for tasks that are more open-ended
  - **Worse for summarization**, as longer output texts are harder to measure
  - **Much worse for dialogue**, which is more open-ended that summarization
  - **Much, much worse story generation**, which is also open-ended, but whose sequence length can make it seem you're getting decent scores!

Failure case of *n*-gram overlap metrics

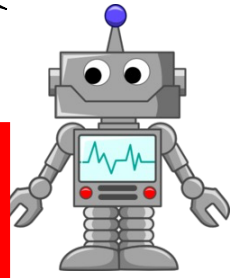
Are you enjoying your school?



Heck yes !

Score:

	0.61	Yes !
	0.25	You know it !
False negative	0	Yup .
False positive	0.67	Heck no !



Fail reason:
does not capture semantic relatedness!

BLEU score

- BLEU: **bi**lingual **e**valuation **u**nderstudy
- Setup: A candidate string $\hat{y}$, a list of reference strings $y^{(1)}, \ldots, y^{(N)}$
- $BLEU(\hat{y}; y^{(1)}, \ldots, y^{(N)})$ should be close to 1 when $\hat{y}$ is similar to $y^{(1)}, \ldots, y^{(N)}$ and close to 0 if not
- Like a teacher trying to score the quality of a student translation $\hat{y}$

Modified n-gram precision

$$p_n(\{\hat{y}\}; \{y\}) = \frac{\sum_{s \in G_n(\hat{y})} \min(C(s, \hat{y}), C(s, y))}{\sum_{s \in G_n(\hat{y})} C(s, \hat{y})}$$

with brevity-penalty $BP(\hat{S}; S) := e^{-(r/c-1)^+}$

$$BLEU_w(\hat{S}; S) := BP(\hat{S}; S) \cdot \exp\left(\sum_{n=1}^{\infty} w_n \ln p_n(\hat{S}; S)\right)$$

Papineni et al., https://aclanthology.org/P02-1040.pdf

ROUGE score

Lin et al., 2004

- **ROUGE**: Recall-Oriented Understudy for Gisting Evaluation
- ROUGE-N: Overlap of n-grams between source and target
  - ROUGE-1: overlap of unigrams
  - ROUGE-2: overlap of bigrams
- ROUGE-L: based on longest common subsequence between source and target

- ***X*** of length *m* (reference)
- ***Y*** of length *n* (translation)

$$R_{lcs} = \frac{LCS(X,Y)}{m} \qquad P_{lcs} = \frac{LCS(X,Y)}{n}$$

$$F_{lcs} = \frac{(1+\beta^2)R_{lcs}P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}}$$

S1. police killed the gunman
S2. police kill the gunman
S3. the gunman kill police

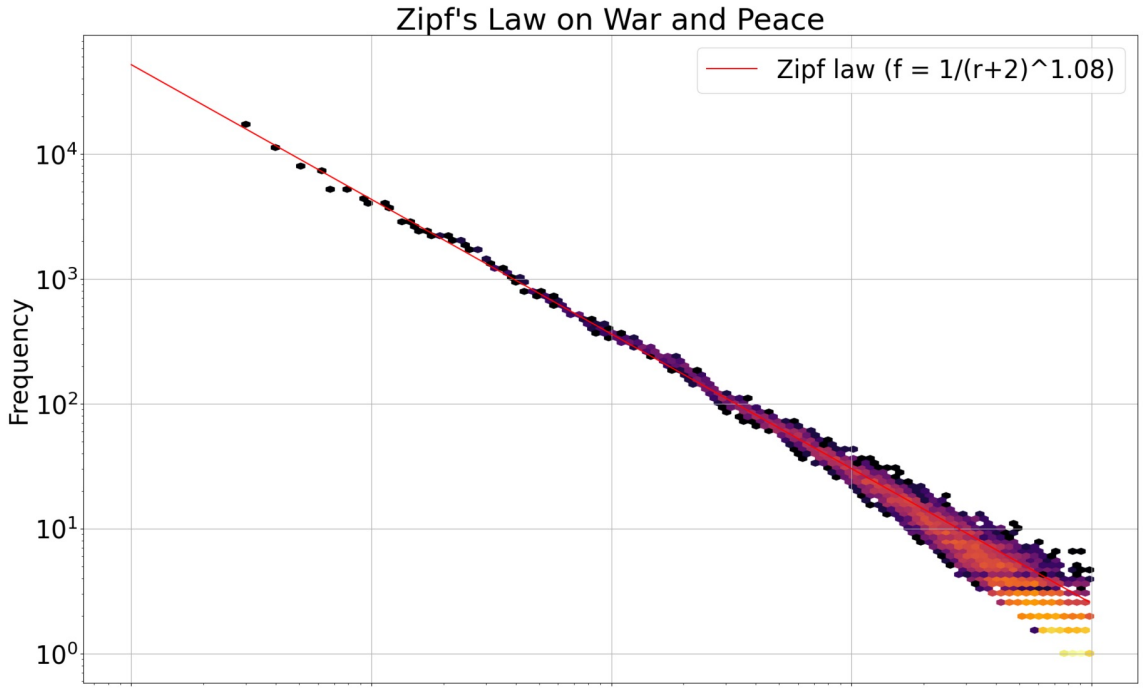
ROUGE-L(S_2) = 0.75

ROUGE-L(S_1) = 0.5

Zipf law

- Zipf's law suggests that there is an exponential relationship between the rank of a word and its frequency in text.

$$\text{frequency} \propto \frac{1}{(\text{rank} + b)^a}$$



Source: https://en.wikipedia.org/wiki/Zipf%27s_law

Zipf's coefficient

- Texts generated by LLMs with different sampling strategies differ from human's in Zipf's coefficient

larger top-k and top-p lead to higher similarity to human text

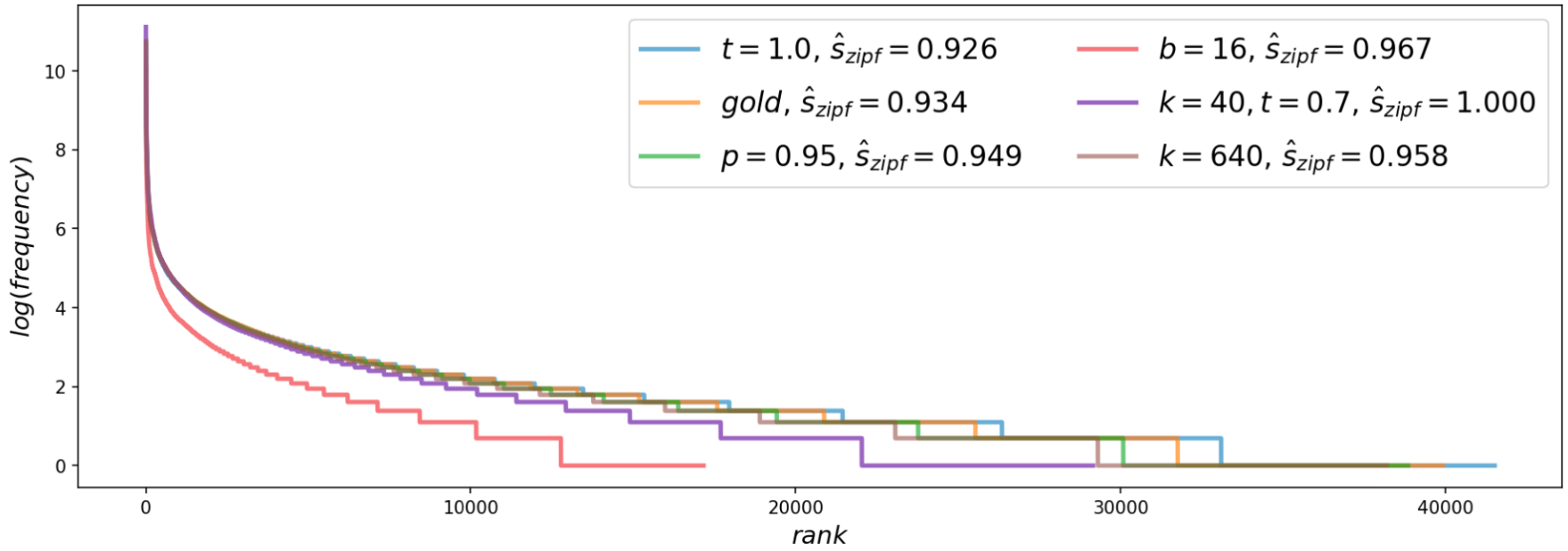
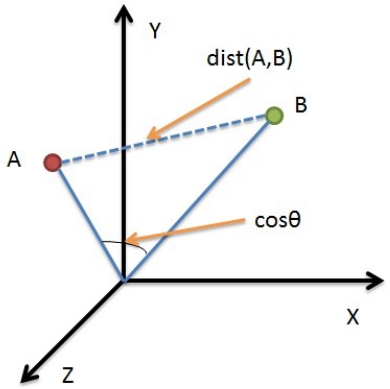


Figure from Holtzman et. al., 2020

Model-based metrics

- Use learned representations of words and sentences to compute semantic similarity between **generated** and **reference** texts
- No more n-gram bottleneck because text units are represented as embeddings!
- The embeddings are pretrained, so the distance metrics used to measure the similarity can be **fixed**
- (the pretrained embeddings are like a static ruler)

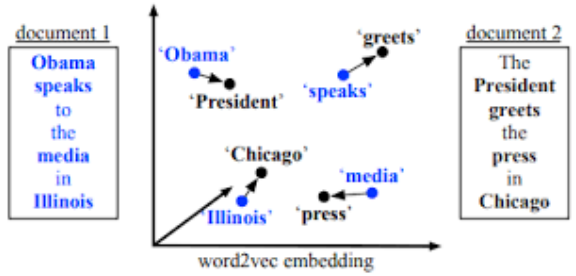
Model-based Metrics: Word level



Vector Similarity

Embedding based similarity for semantic distance between text.

- **Embedding Average** (Liu et al., 2016)
- **Vector Extrema** (Liu et al., 2016)
- **MEANT** (Lo, 2017)
- **YISI** (Lo, 2019)



Word Mover's Distance

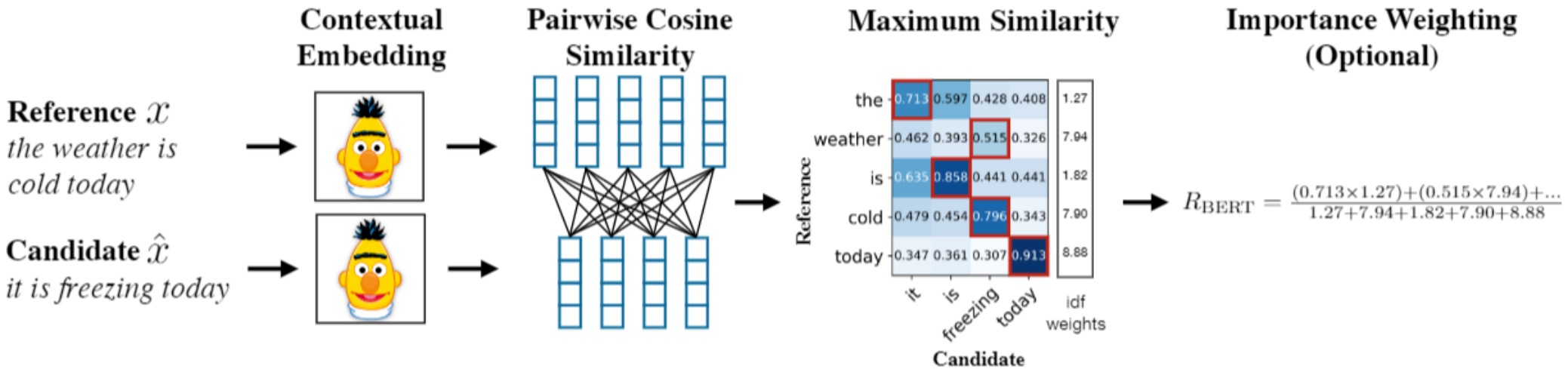
Measures the distance between two sequences (e.g., sentences, paragraphs, etc.), using word embedding similarity matching.
(Kusner et.al., 2015; Zhao et al., 2019)

Slide credit to: https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1244/

Model-based Metrics: Word level

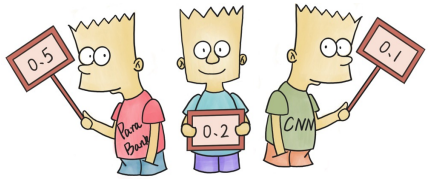
BERTScore

Uses pre-trained contextual embeddings from BERT and matches words in candidate and reference sentences by cosine similarity.



Zhang et. al., 2020

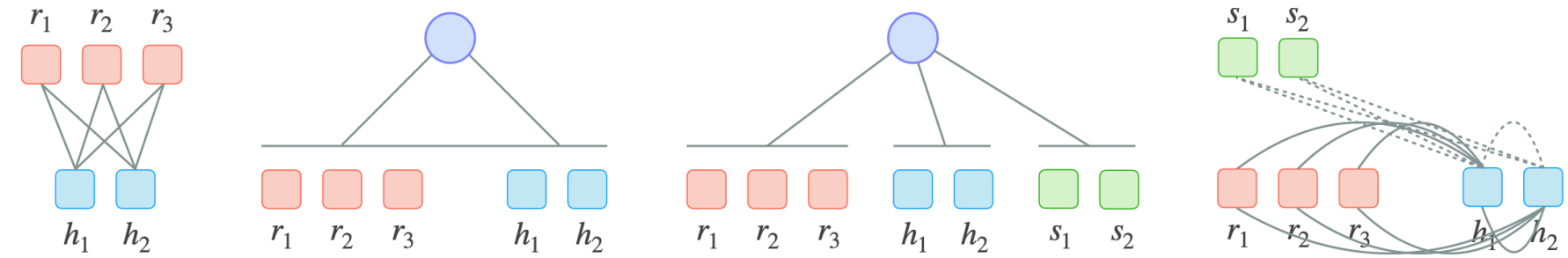
BARTScore



Generate target y from source x

$$p(\mathbf{y}|\mathbf{x},\theta) = \prod_{t=1}^{m} p(\mathbf{y}_t|\mathbf{y}_{<t},\mathbf{x},\theta)$$

- Evaluating generated text as text generation



(a) Matching (b) Regression (c) Ranking (d) Generation

BERTScore is here

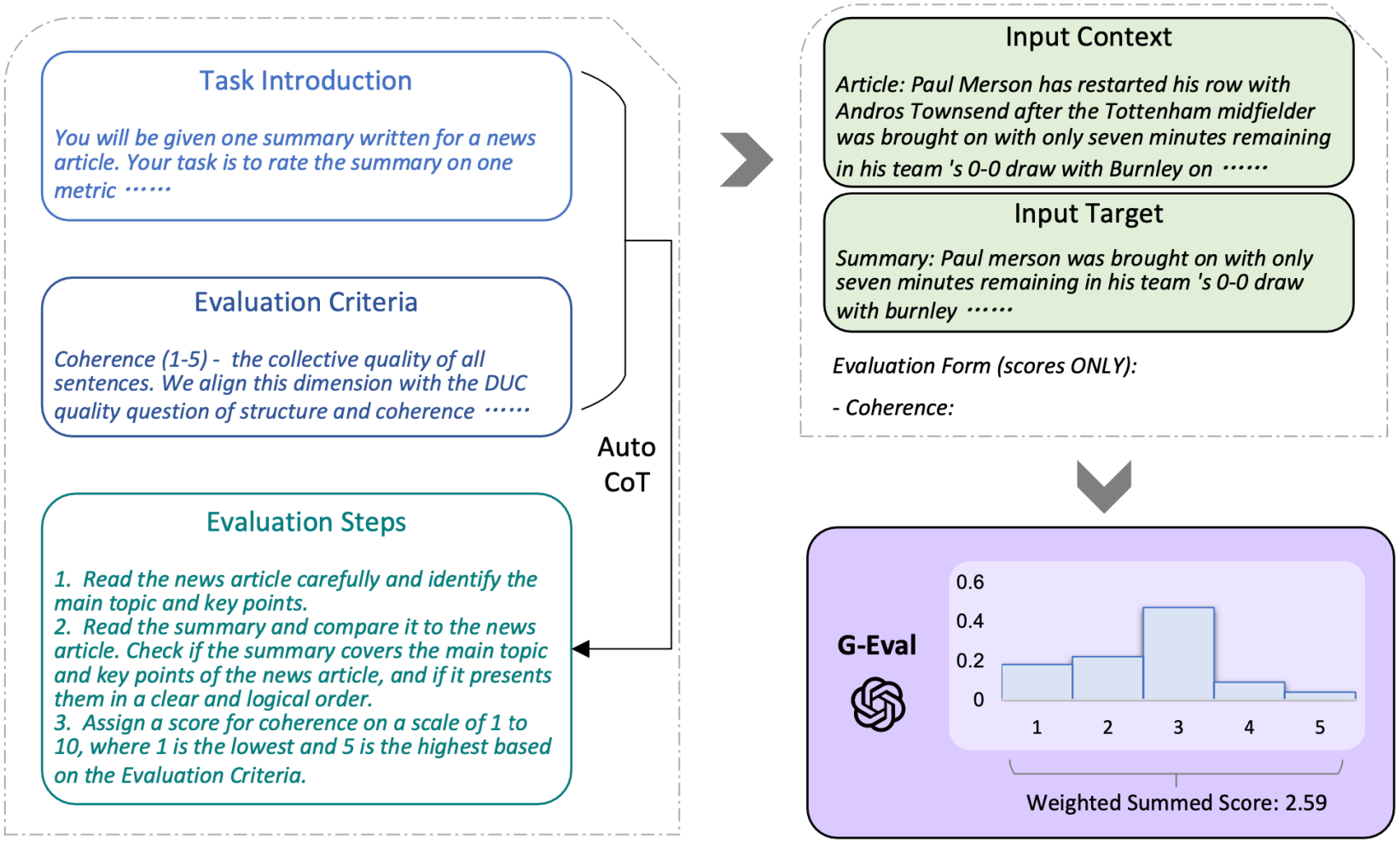
BARTScore is the weighted log likelihood of generation

$$\text{BARTSCORE} = \sum_{t=1}^{m} \omega_t \log p(\mathbf{y}_t|\mathbf{y}_{<t},\mathbf{x},\theta)$$

Yuan et al. 2021, https://github.com/neulab/BARTScore

G-Eval

NLG evaluation using GPT-4, Liu et al. 2013



G-eval with GPT-4 achieves high correlation with human on summarization task

outperforming other automatic/model-based metrics

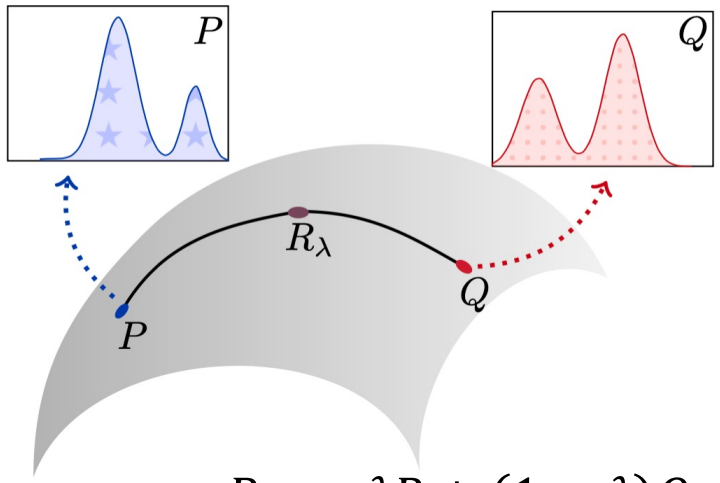
Evaluation at document-level

Pillutla et.al., 2022

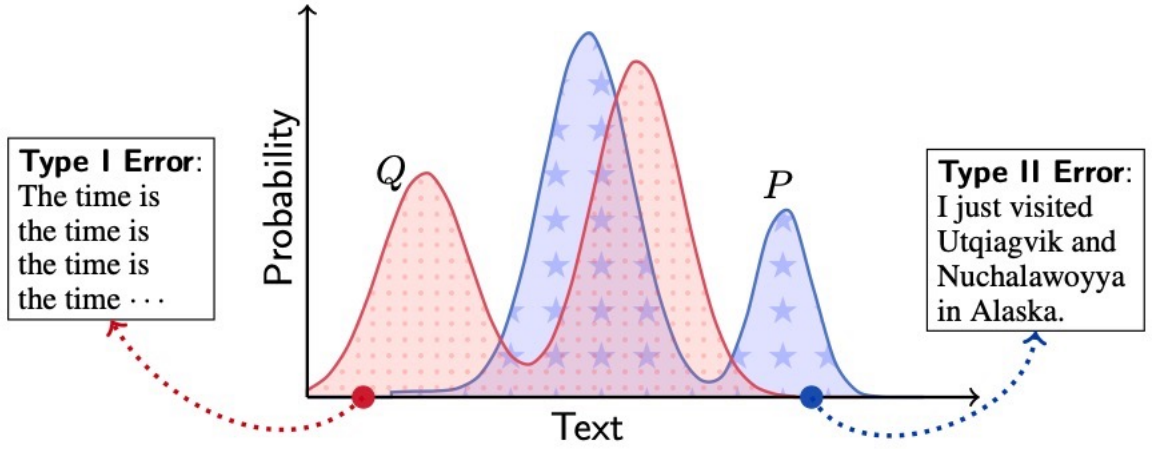
- **MAUVE**: computes information divergence in a quantized embedding space, between the generated text and the gold reference text.

P: human text

Q: machine text



$$R_\lambda = \lambda P + (1-\lambda)Q,$$
$$\lambda \in (0,1)$$



MAUVE (details)

Represent text with quantized clusters in embedding space

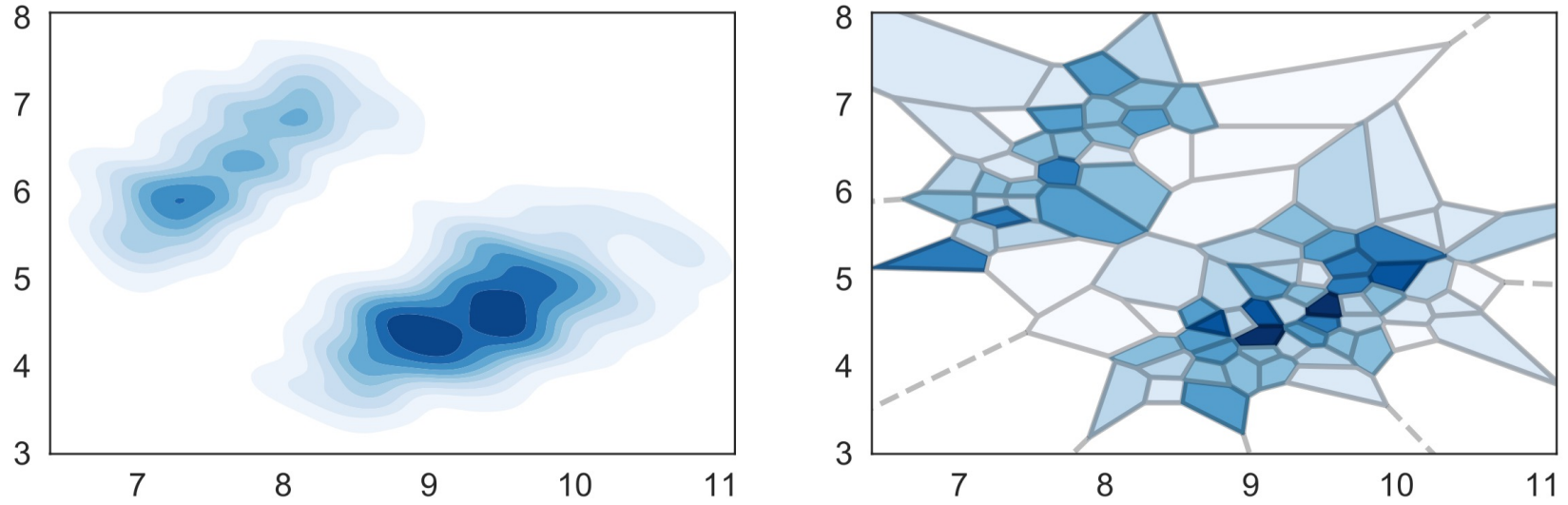
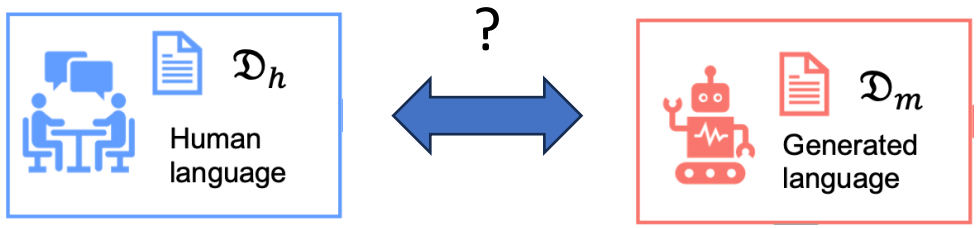


Figure 3: Illustration of the quantization. **Left**: A continuous two-dimensional distribution P. **Right**: A partitioning of the Euclidean plane $\mathbb{R}^2$ and the corresponding quantized distribution $\tilde{P}$.

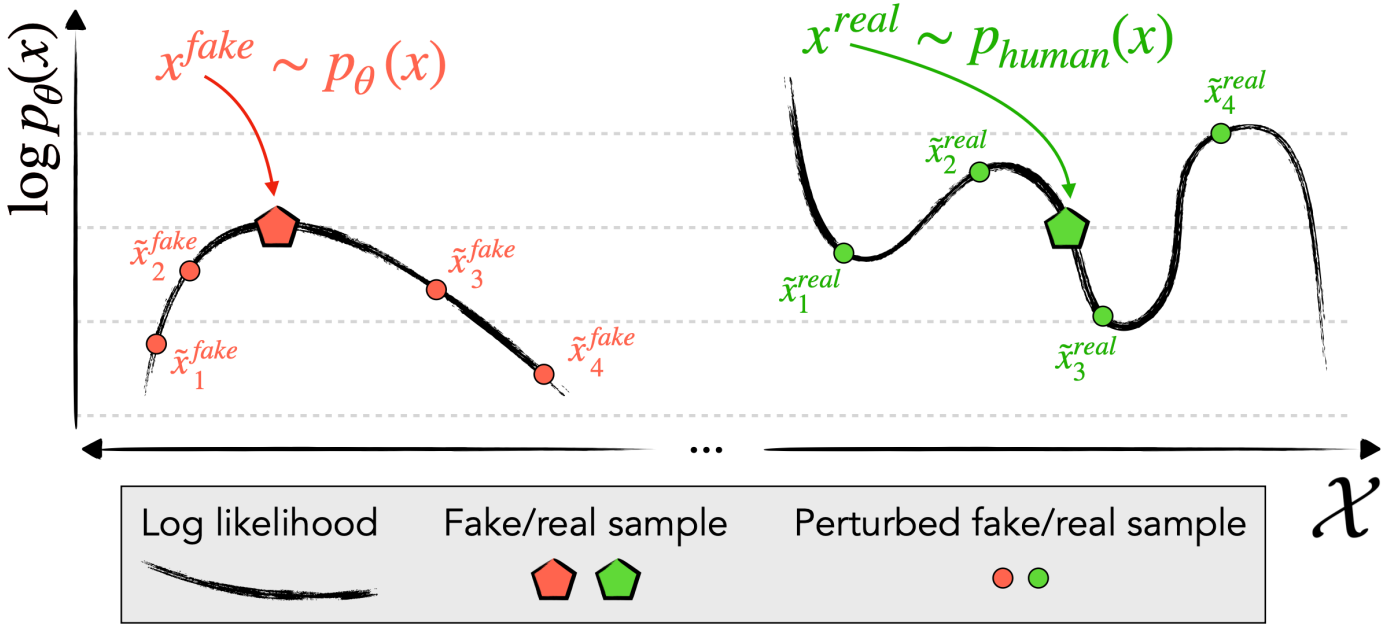
Evaluation => Detection

- How do we distinguish LLM-generated content from real human-generated one?
- What are the differences?



Detecting difference in likelihood space

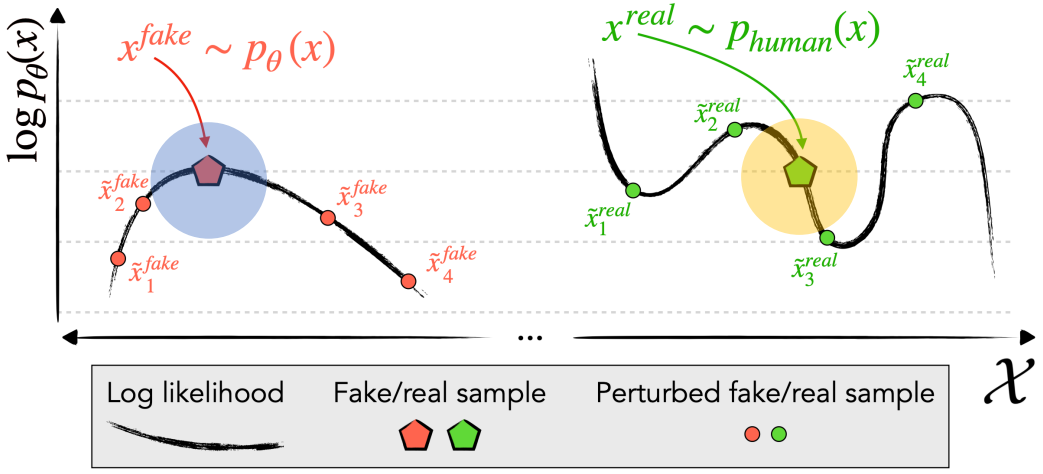
- DetectGPT (Mitchell et al., 2023) and Faster-DetectGPT (Bao et al., 2024)

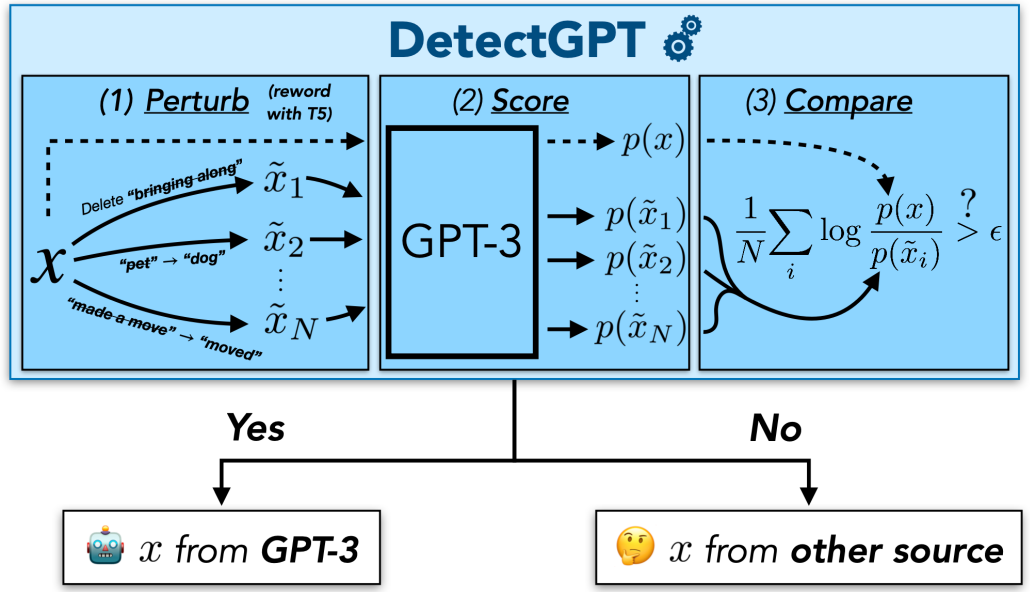


Unlike human-written text, model-generated text tends to lie in areas where the log probability function has negative curvature (曲率) (e.g., local maxima of the log probability)

Model based metrics: Go Beyond Semantics

- DetectGPT (Mitchell et al., 2023) and Faster-DetectGPT (Bao et al., 2024)





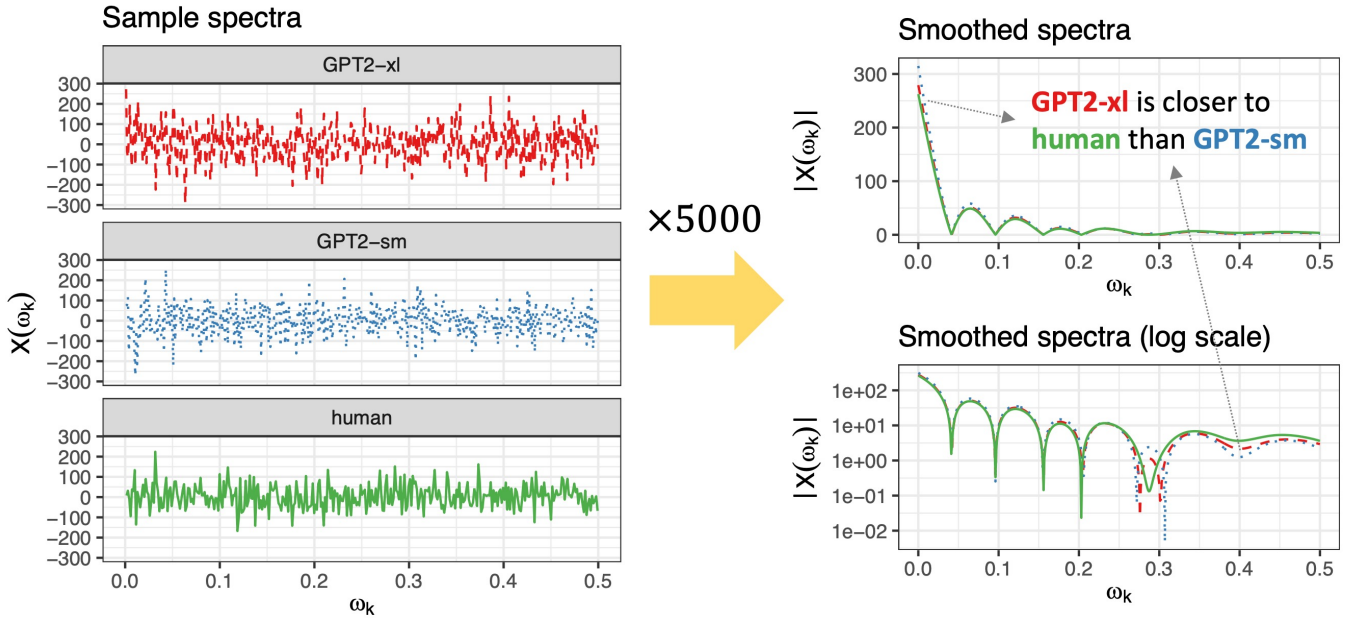
If we perturb an already maximized text, it has good chance to go worse

If we perturb a text not "optimized", it has equal chances of going worse or better

Model based metrics: Go Beyond Semantics

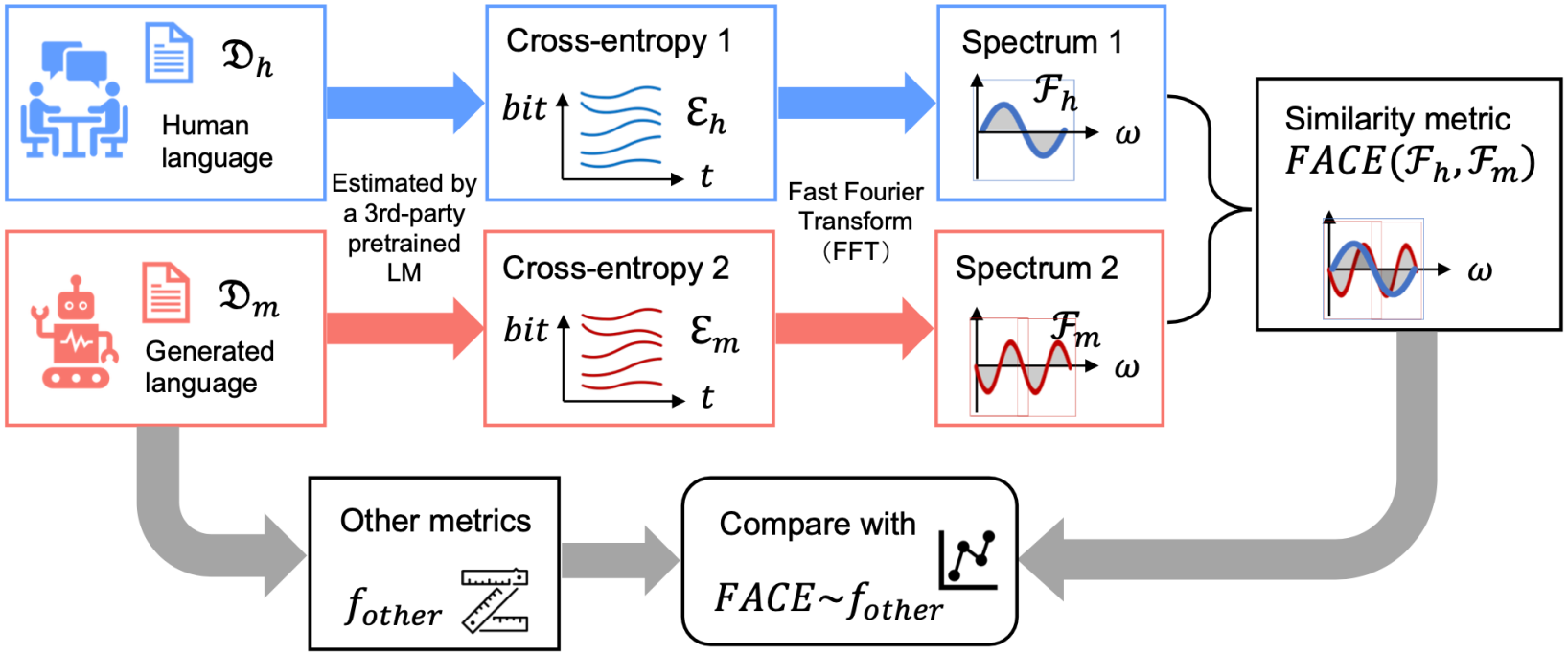
- **FACE: Fourier Analysis of Cross-Entropy** (Yang et al., 2023)
- Key questions: Why is the log probability of language changing in human language? How does it change?

Assumption: There is a frequency component (not easily detected) in natural language, which may not be learned by models

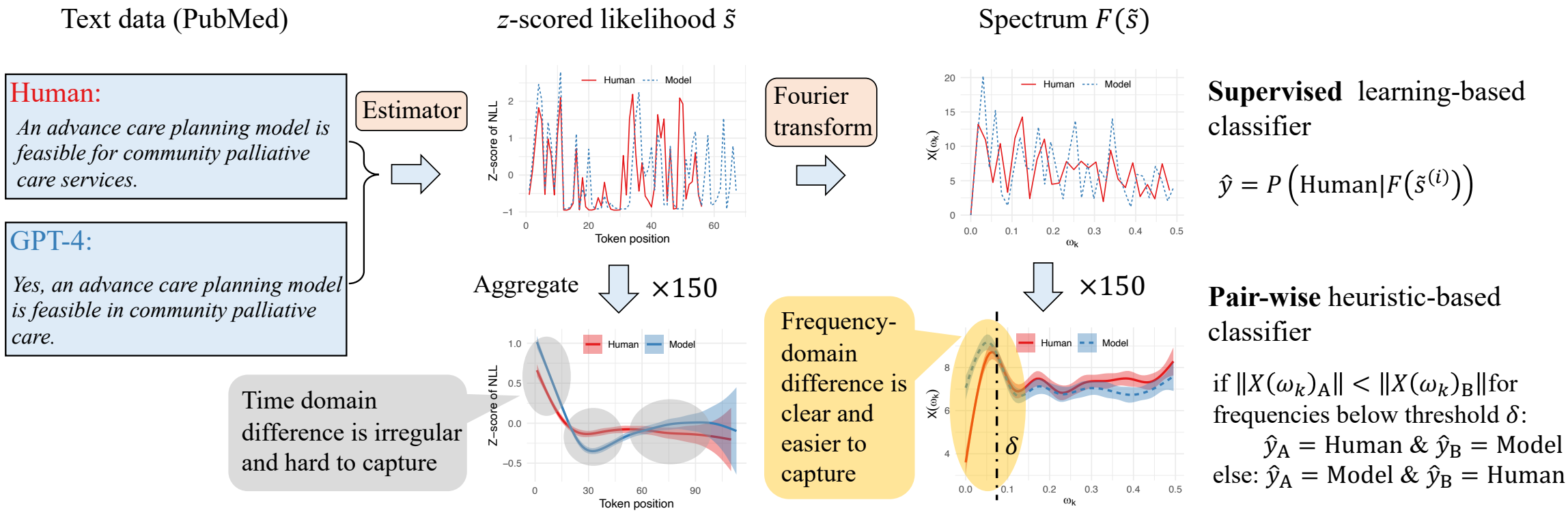


Model based metrics: Go Beyond Semantics

- **FACE: Fourier Analysis of Cross-Entropy** (Yang et al., 2023)



FourierGPT: Detecting subtle differences



Supervised learning-based classifier

$$\hat{y} = P\left(\text{Human} | F\left(\tilde{s}^{(i)}\right)\right)$$

Pair-wise heuristic-based classifier

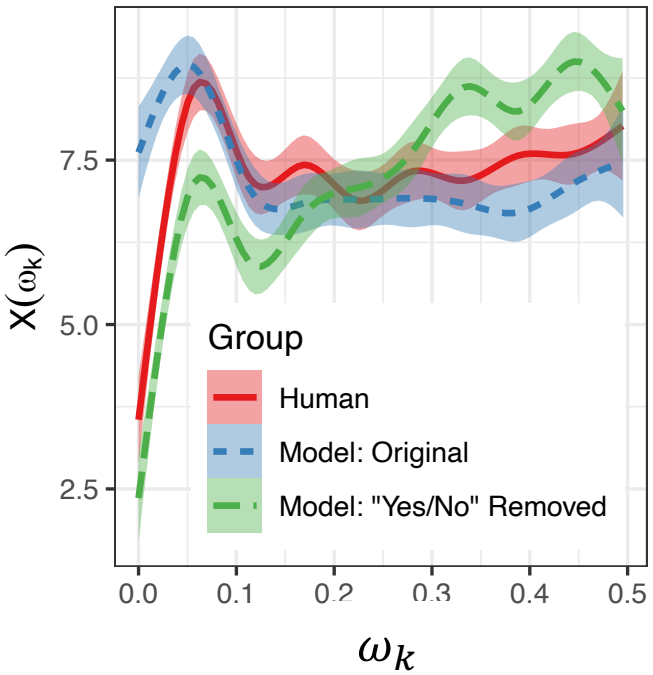
if $\|X(\omega_k)_A\| < \|X(\omega_k)_B\|$ for frequencies below threshold δ:

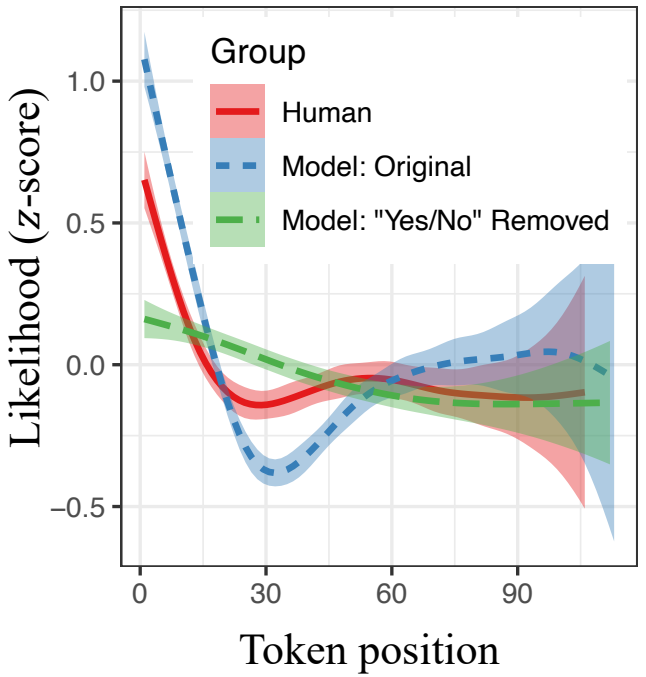
$\hat{y}_A = \text{Human}$ & $\hat{y}_B = \text{Model}$

else: $\hat{y}_A = \text{Model}$ & $\hat{y}_B = \text{Human}$

Xu et al. 2024, https://aclanthology.org/2024.emnlp-main.564/

FourierGPT: Detecting subtle differences





- Models tend to answer with “Yes” or “No”
- Humans do not

Xu et al. 2024, https://aclanthology.org/2024.emnlp-main.564/

Human Evaluations

- Automatic metrics fall short of matching human decisions
- Human evaluation is most important form of evaluation for text generation systems.
- Gold standard in developing new automatic metrics:
  - New automated metrics must correlate well with human evaluations!

Human Evaluations

Human Evaluations

- **Ask humans** to evaluate the quality of generated text
- Overall or along some specific dimension:
- fluency
- coherence / consistency
- factuality and correctness
- commonsense
- style / formality
- grammaticality
- typicality
- redundancy

Human Evaluation: Example

- Question a: "Which continuation is more interesting or creative, given the context?"
- Question b: "Which continuation makes more sense, given the context?"
- Question c: "Which continuation is more likely to be written by a human?"

Like an extremely simplified Turing test

Sampled from the data provided by Pillutla et.al., 2022

MT-Bench human annotation

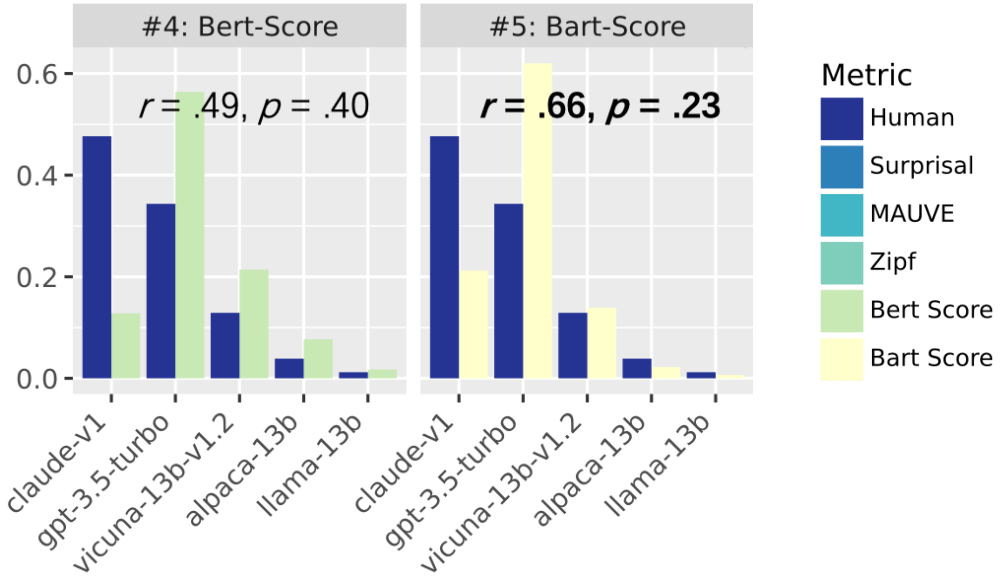
question_id int64 81 – 160	model_a string · classes 6 values	model_b string · classes 6 values	winner string · classes 3 values	judge string · lengths 8 – 9	conversation_a list	conversa list
81	alpaca-13b	gpt-3.5-turbo	model_b	author_2	[{ "content": "Compose an engaging travel blo…	[{ "con an engag
81	alpaca-13b	gpt-3.5-turbo	model_b	author_2	[{ "content": "Compose an engaging travel blo…	[{ "con an engag

Meaning: GPT-3.5 beats Alpaca-13b, over the generation on Question 81, according to Judge “author_2”

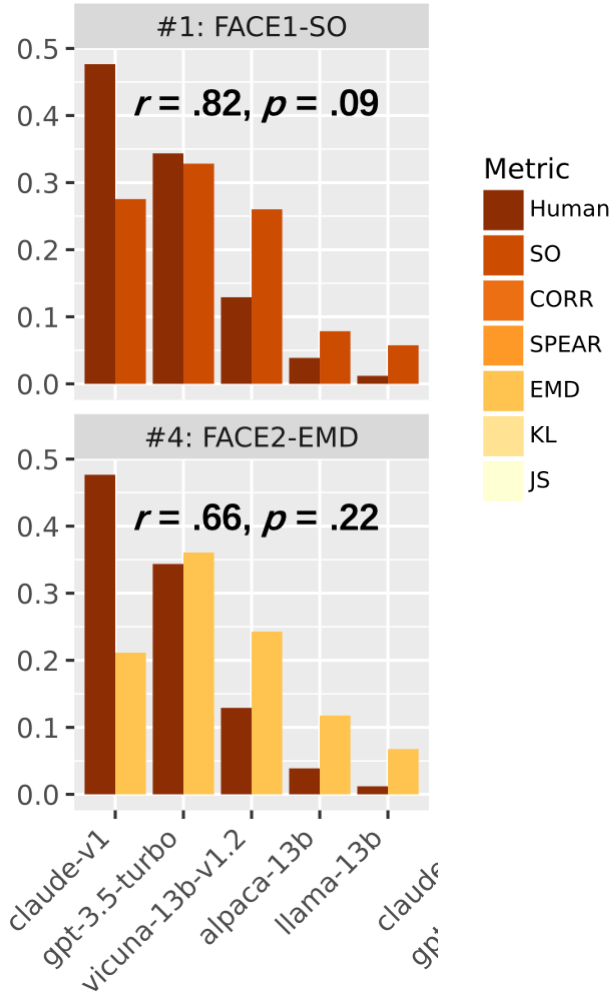
Source: https://huggingface.co/datasets/lmsys/mt_bench_human_judgments

Pair-wise human preferences aggregated

- Using Bradley-Terry algorithm to sort models



Some metrics produce better alignment with human judges



According **Human**: claude-v1 > gpt-3.5 > vicuna-13b-v1 > …

According **BERTScore**: gpt-3.5 > vicuna-13b-v1 > claude-v1 > …

According **FACE-SO**: gpt-3.5 > claude-v1 > vicuna-13b-v1 > …

Similar method to rate LLMs in generation quality

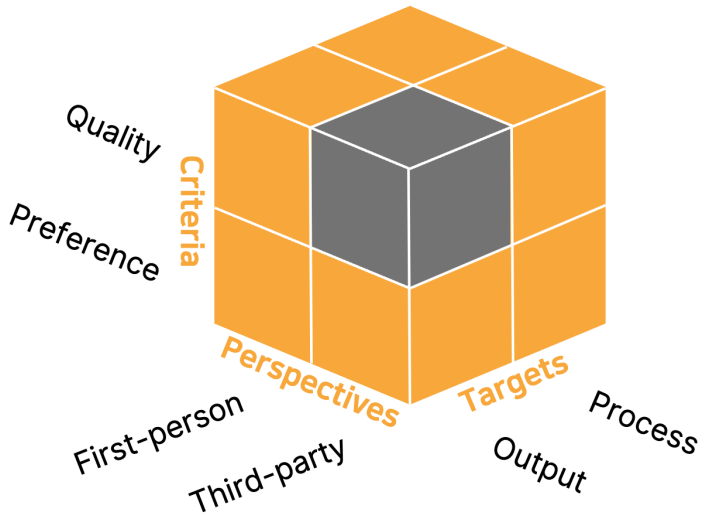
- lmarena.ai

🏆 Overall | Search by model name... / | Default

Rank (UB) ↑	Rank (Style Control) ↑↓	Model ↑↓	Score ↑↓	95% CI (±) ↑↓	Votes ↑↓	Organization ↑↓	License ↑↓
1	1	gemini-2.5-pro-exp-03-25	1439	+6/-5	10,389	Google	Proprietary
2	1 ↑	o3-2025-04-16	1418	+14/-9	2,211	OpenAI	Proprietary
2	2	chatgpt-4o-latest-20250326	1408	+6/-5	9,229	OpenAI	Proprietary
3	5 ↓	grok-3-preview-02-24	1402	+4/-5	14,840	xAI	Proprietary
3	5 ↓	gemini-2.5-flash-preview-04-17	1393	+10/-7	4,073	Google	Proprietary
4	3 ↑	gpt-4.5-preview-2025-02-27	1398	+4/-5	15,285	OpenAI	Proprietary
7	11 ↓	gemini-2.0-flash-thinking-exp-01-21	1380	+4/-4	26,903	Google	Proprietary
7	5 ↑	deepseek-v3-0324	1373	+6/-7	6,792	DeepSeek	MIT
8	5 ↑	gpt-4.1-2025-04-14	1363	+10/-9	2,927	OpenAI	Proprietary
9	7 ↑	deepseek-r1	1358	+5/-4	16,857	DeepSeek	MIT

Evaluate LMs by Interacting with Them

- Evaluating Human-Language Model Interaction (Lee et al., 2022)



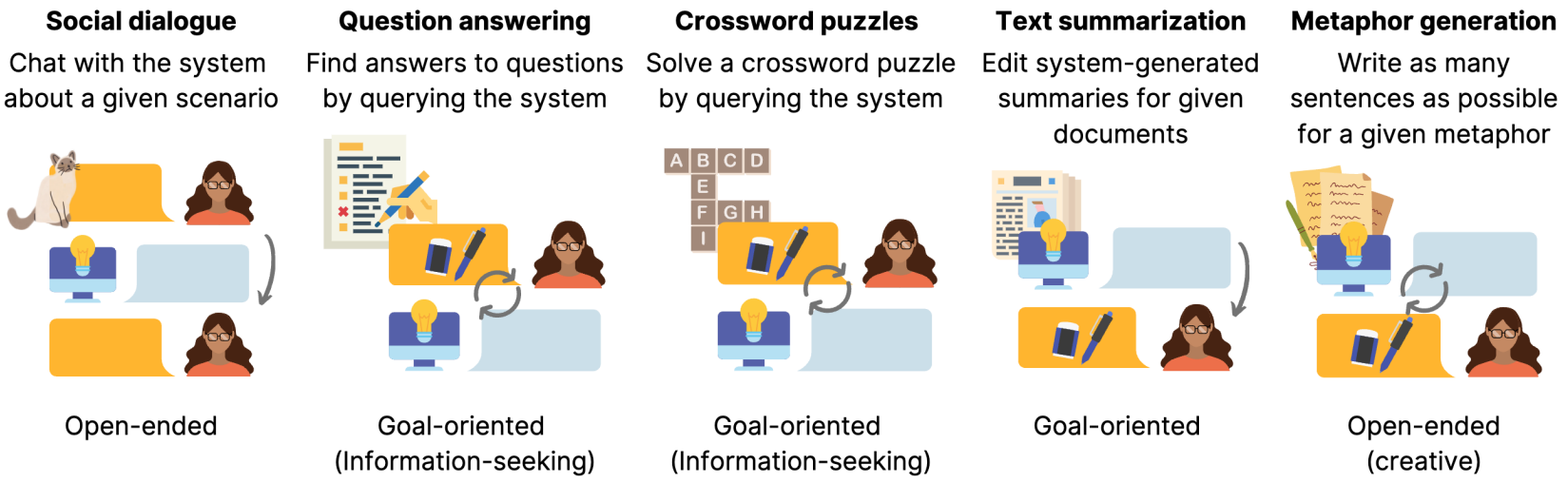


Figure 2: **Five tasks and human-LM interaction in the context of accomplishing the tasks.** We

Human evaluation: Issues

- Human judgments are regarded as the **gold standard**
- Human eval is **slow** and **expensive**

- Beyond the cost, it's still far from perfect:
- Results are inconsistent/not reproducible
- can be illogical
- misinterpret your question

Recep for Evaluation

- **Content overlap metrics** provide a good starting point, but they're not good enough on their own.
- **Model-based metrics** can be more correlated with human judgment, but behavior is not interpretable
- **Human judgments** are critical, but humans are inconsistent!

- Suggestions:
- Look at your model generations. Don't just rely on numbers!
- Publicly release large samples of the output of systems that you create

References

- Holtzman, A., Buys, J., Du, L., Forbes, M., & Choi, Y. (2019). The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.
- Pillutla, K., Swayamdipta, S., Zellers, R., Thickstun, J., Welleck, S., Choi, Y., & Harchaoui, Z. (2021). Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, *34*, 4816-4828.
- Su, Y., Lan, T., Wang, Y., Yogatama, D., Kong, L., & Collier, N. (2022). A contrastive framework for neural text generation. *Advances in Neural Information Processing Systems*, *35*, 21548-21561.
- Li, X. L., Holtzman, A., Fried, D., Liang, P., Eisner, J., Hashimoto, T., ... & Lewis, M. (2022). Contrastive decoding: Open-ended text generation as optimization. *arXiv preprint arXiv:2210.15097*.
- Lee, M., Srivastava, M., Hardy, A., Thickstun, J., Durmus, E., Paranjape, A., ... & Liang, P. (2022). Evaluating human-language model interaction. *arXiv preprint arXiv:2212.09746*.
- Mitchell, E., Lee, Y., Khazatsky, A., Manning, C. D., & Finn, C. (2023, July). Detectgpt: Zero-shot machine-generated text detection using probability curvature. In *International Conference on Machine Learning* (pp. 24950-24962). PMLR.
- Yang, Z., Yuan, Y., Xu, Y., Zhan, S., Bai, H., & Chen, K. (2024). Face: Evaluating natural language generation with fourier analysis of cross-entropy. *Advances in Neural Information Processing Systems*, *36*.
- Token Sampling Methods, Distilled AI, https://aman.ai/primers/ai/token-sampling/