



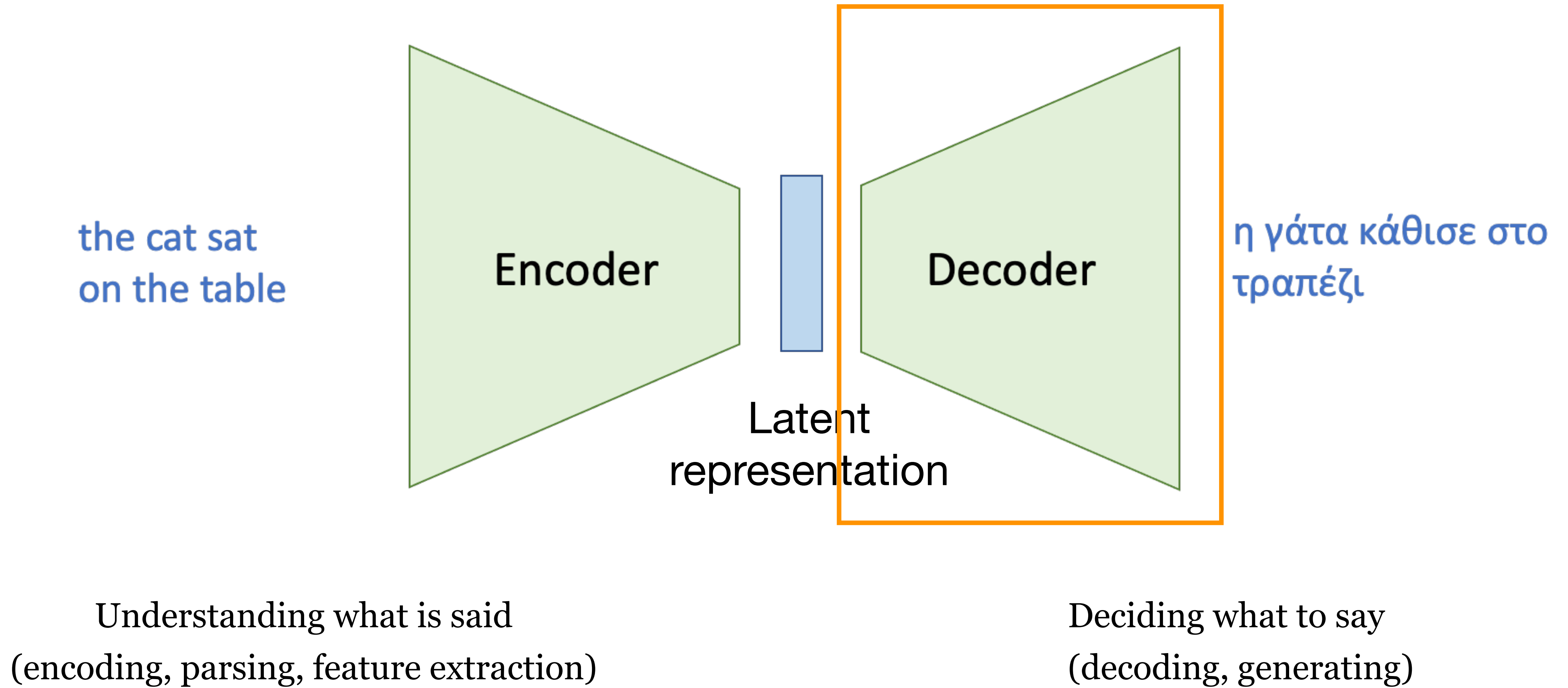
CMPT 713: Natural Language Processing

Text Generation

Spring 2023
2023-03-29

Adapted from slides from Chris Manning and Antoine Bosselut
(with some content from slides from Graham Neubig)

Encoder-Decoder Model

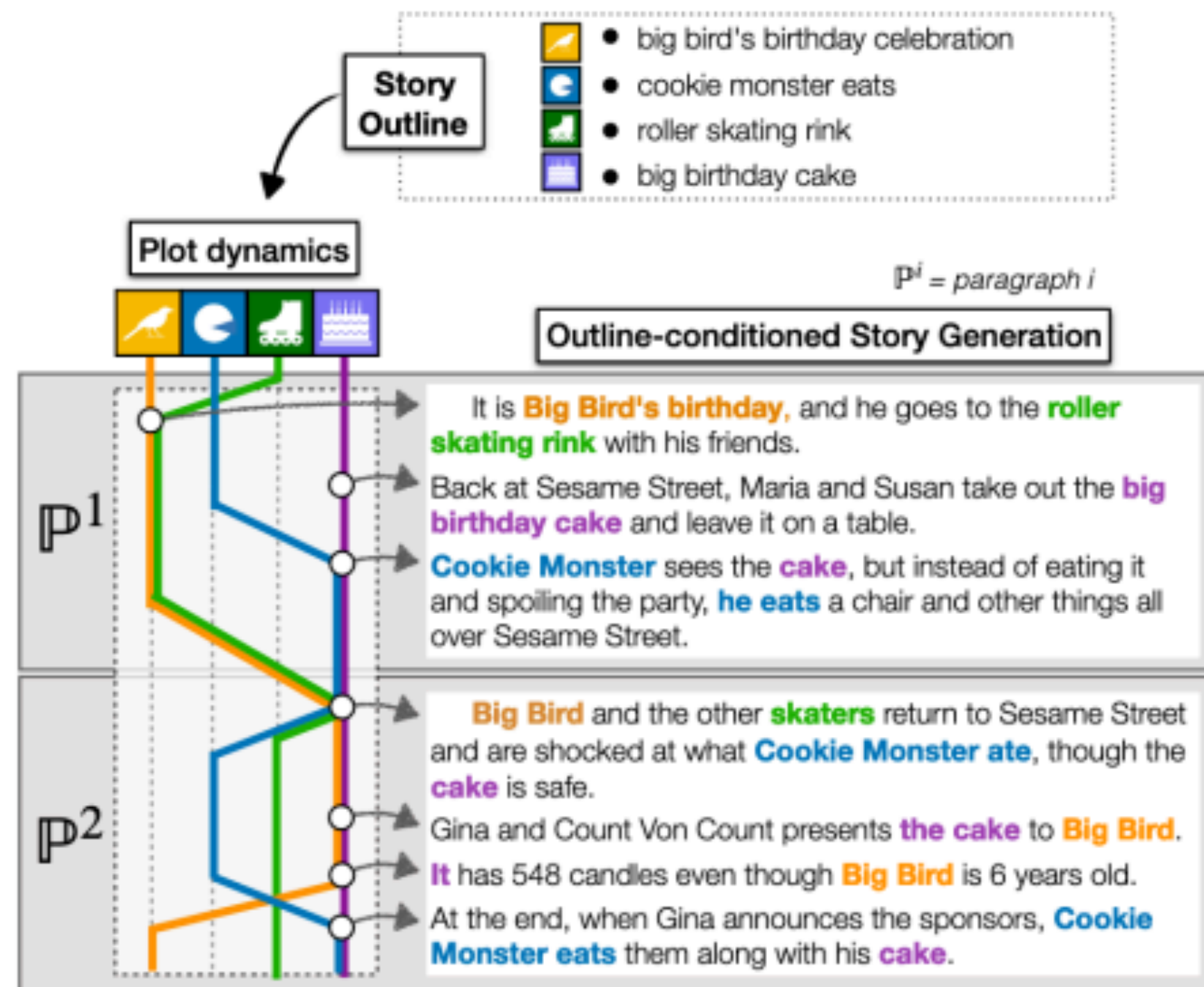


Many tasks and applications for natural language generation (NLG)

Task/Application	Input	Output
Machine Translation	French	English
Summarization	Document	Short Summary
Dialogue	Utterance Dialog history	Response
Image Captioning	Image	Caption
Story Generation	Prompt	Story

Examples of NLG

Creative stories



(Rashkin et al., EMNLP 2020)

Data-to-text

Table Title: Robert Craig (American football)
Section Title: National Football League statistics
Table Description: None

YEAR	TEAM	RUSHING					RECEIVING				
		ATT	YDS	AVG	LNG	TD	NO.	YDS	AVG	LNG	TD
1983	SF	176	725	4.1	71	8	48	427	8.9	23	4
1984	SF	155	649	4.2	28	4	71	675	9.5	64	3
1985	SF	214	1050	4.9	62	9	92	1016	11	73	6
1986	SF	204	830	4.1	25	7	81	624	7.7	48	0
1987	SF	215	815	3.8	25	3	66	492	7.5	35	1
1988	SF	310	1502	4.8	46	9	76	534	7.0	22	1
1989	SF	271	1054	3.9	27	6	49	473	9.7	44	1
1990	SF	141	439	3.1	26	1	25	201	8.0	31	0
1991	RAI	162	590	3.6	15	1	17	136	8.0	20	0
1992	MIN	105	416	4.0	21	4	22	164	7.5	22	0
1993	MIN	38	119	3.1	11	1	19	169	8.9	31	1
Totals	-	1991	8189	4.1	71	56	566	4911	8.7	73	17

Craig finished his eleven NFL seasons with 8,189 rushing yards and 566 receptions for 4,911 receiving yards.

(Parikh et al., EMNLP 2020)

Visual description



Two children are sitting at a table in a restaurant. The children are one little girl and one little boy. The little girl is eating a pink frosted donut with white icing lines on top of it. The girl has blonde hair and is wearing a green jacket with a black long sleeve shirt underneath. The little boy is wearing a black zip up jacket and is holding his finger to his lip but is not eating. A metal napkin dispenser is in between them at the table. The wall next to them is white brick. Two adults are on the other side of the short white brick wall. The room has white circular lights on the ceiling and a large window in the front of the restaurant. It is daylight outside.

(Krause et al. CVPR 2017)

ChatGPT: a SoTA NLG system

Idea generation



Got any creative ideas for a 10 year old's birthday?



Here are some creative birthday ideas for a 10 year old:

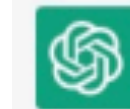


1. Scavenger Hunt: A birthday scavenger hunt can be a fun and interactive activity for kids. You can create a series of clues that lead the birthday child to the next one, until they reach the final prize.
2. DIY Crafts: Crafts are always a hit with kids. You can arrange a crafting session where the birthday child and their friends can make their own personalized gifts or decorations.
3. Movie Night: Have a movie night with friends and family at home or at a local theater. You can also make it a sleepover by setting up sleeping bags, pillows, and blankets.

Poem generation



Write me a poem about sorting algorithms



Sorting algorithms, oh so wise,
Bringing order to chaotic ties.
Bubble sort, quick sort, merge sort too,
Each one unique, with their own special view.

Bubble sort, gentle as can be,
Compares and swaps, so efficiently.
It moves like waves, across the sea,
Bringing order, so beautifully.

Quick sort, so fast and strong,
Divides and conquers all day long.
With each partition, it grows so wide,
Order is found, on every side.

Merge sort, so neat and trim,
Divides and merges, one by one.
It builds a bridge, to conquer all,
Order is its overall call.

Spectrum of NLG tasks



Machine Translation

Source Sentence: 当局已经宣布今天是节假日。

Reference Translations:

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 not very diverse

Spectrum of NLG tasks



ChitChat Dialog

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.

More possible “correct” generations

Spectrum of NLG tasks



Story Generation

Input: Write a story about three little pigs?

Outputs:

... (lots of different options!)

Very open-ended!

Spectrum of NLG tasks

Less open-ended

More open-ended



Machine
Translation

Summarization

Task-driven
Dialog

ChitChat
Dialog

Story
Generation

Less diverse

More diverse

Output is mostly
determined by the input

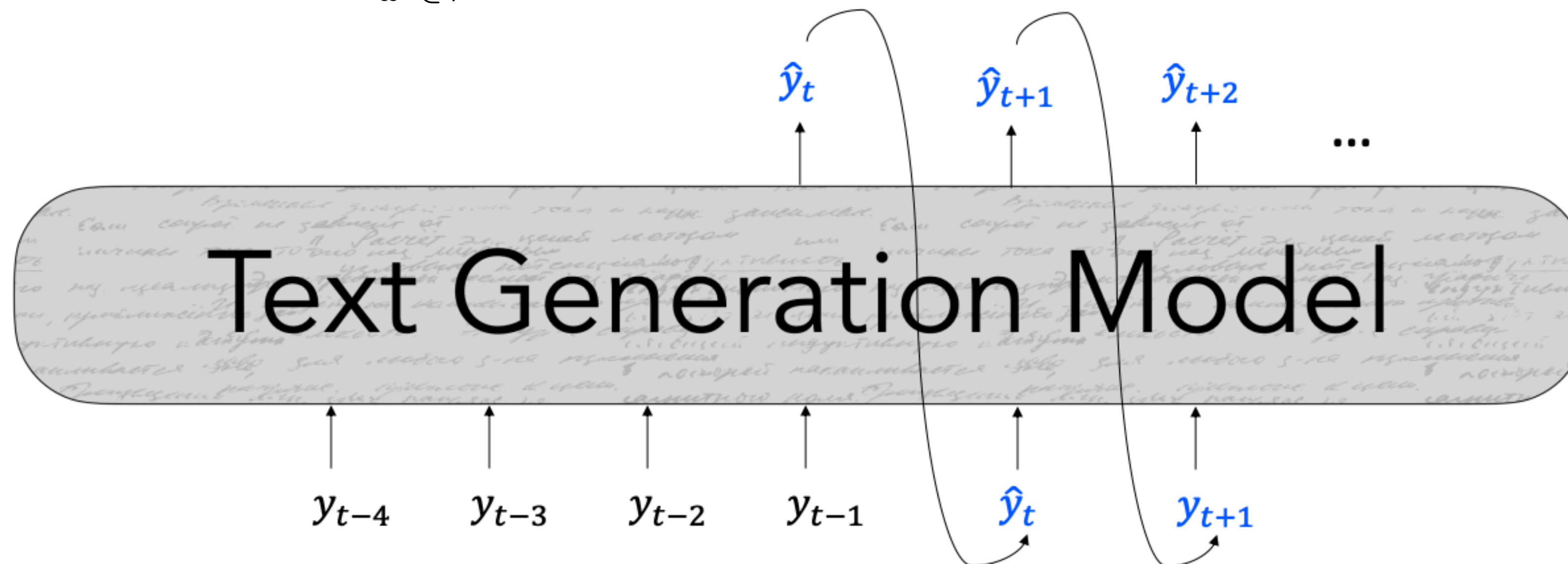
Lots of freedom in the
output, output distribution
should be varied and
diverse

Can characterize the spectrum of tasks using **entropy**.
Can use different **decoding and training strategies** for each.

Review of autoregressive text generation

- Autoregressive text models generate future words based on past words
- At each time step t , the model is given sequence of tokens as input $\{y\}_{<t}$ and predicts next token \hat{y}_t
- For model $f(\cdot)$ and vocabulary V , the model estimate the probability of the next token by taking the softmax of the scores: $S = f(\{y_{<t}, \theta\}) \in \mathbb{R}^V$

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

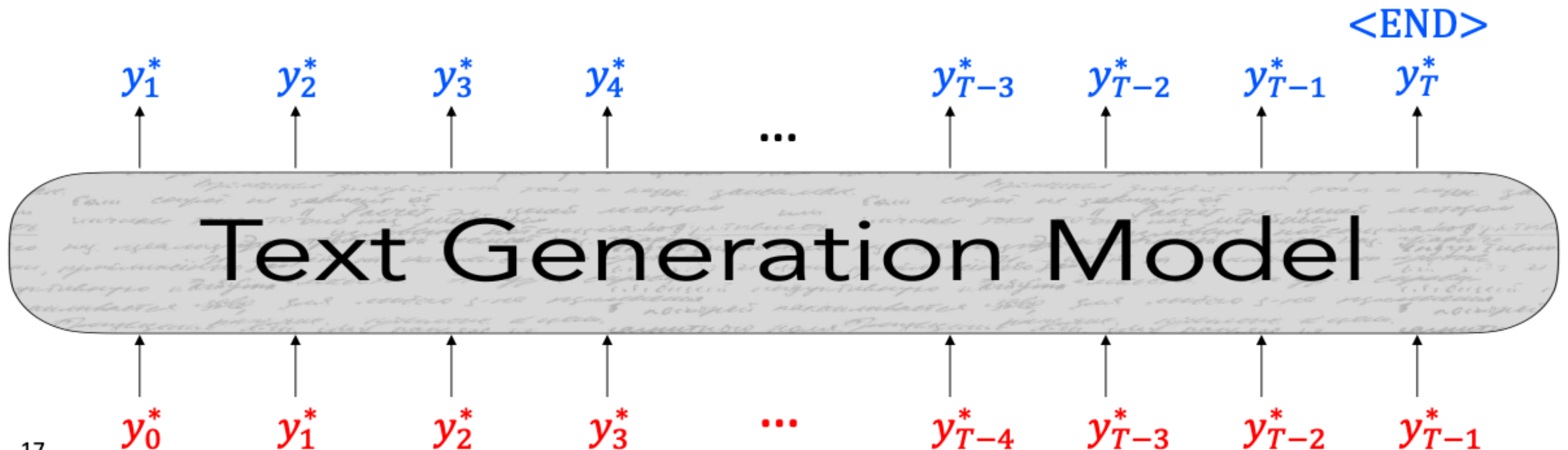


Training with maximum likelihood

- Trained to maximize the probability of the next token y_t^* given preceding words $\{y_{<t}^*\}$

$$\mathcal{L}_{MLE} = - \sum_{t=1}^T \log P(y_t^* | \{y_{<t}^*\})$$

- Classification task over the vocabulary at each time step
- Training with the ground-truth is also known as “teacher forcing”



How to predict the next word?

- At each time step t , our **model** f computes a vector of scores for each token in our vocabulary: $S \in \mathbb{R}^{|V|}$

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

- Then, we compute a probability distribution P_t over these scores (usually with a softmax function):

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

- Our **decoding** algorithm then defines a function g to select a token from this distribution:

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

Word prediction

Handling large vocabularies

- ▶ Softmax can be expensive for large vocabularies

$$P(y_i) = \frac{\exp(w_i \cdot h + b_i)}{\sum_{j=1}^{|V|} \exp(w_j \cdot h + b_j)}$$

← Expensive to compute

- ▶ English vocabulary size: 10K to 100K

Negative Sampling

- Softmax is expensive when vocabulary size is large

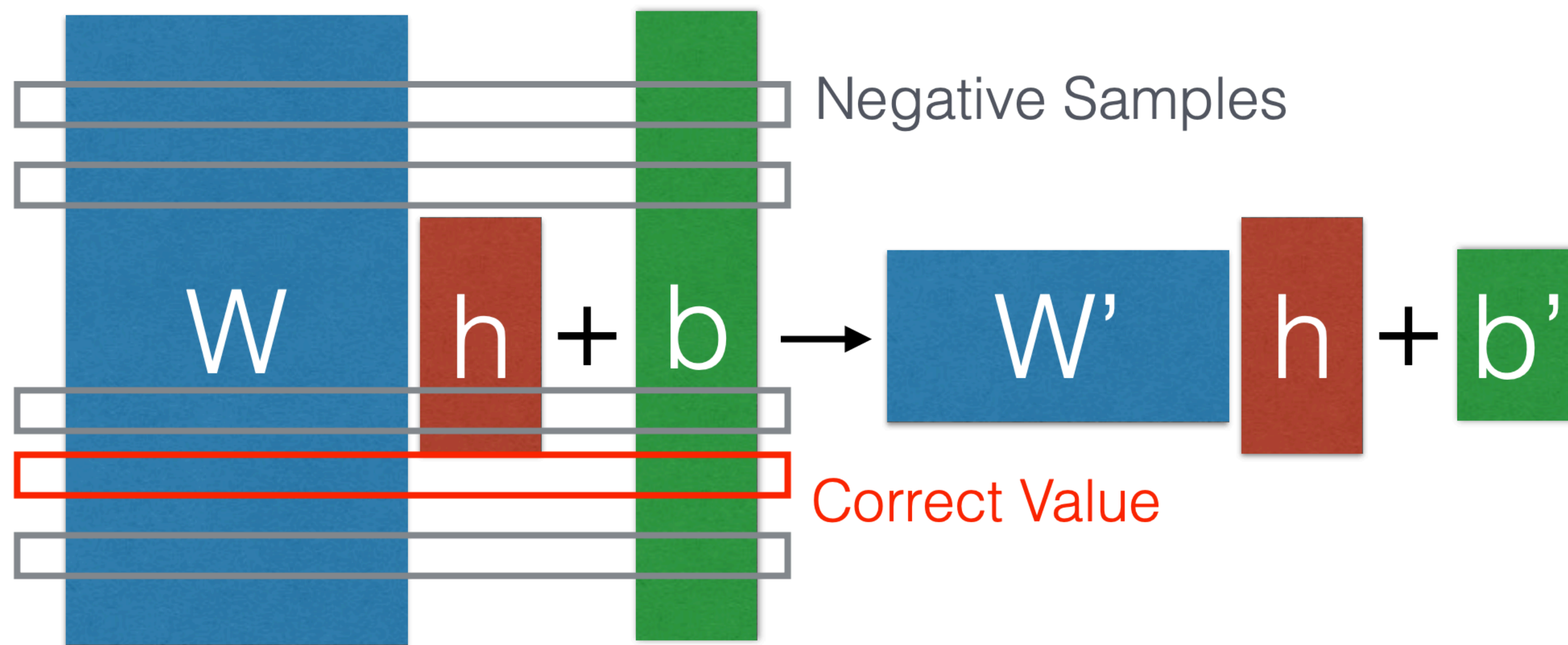


The diagram illustrates the Softmax operation in a neural network layer. It features three vertical bars: a light gray bar on the left labeled 'p', a large blue bar in the middle labeled 'W', a small red bar to the right of 'W' labeled 'h', and a green bar on the far right labeled 'b'. The equation $p = \text{softmax}(Wh + b)$ is displayed between the bars, with the 'h' and '+' sign positioned between the red and green bars respectively.

$$p = \text{softmax}(Wh + b)$$

Negative Sampling

- Sample just a subset of the vocabulary for negative
- Saw simple negative sampling in word2vec (Mikolov 2013)



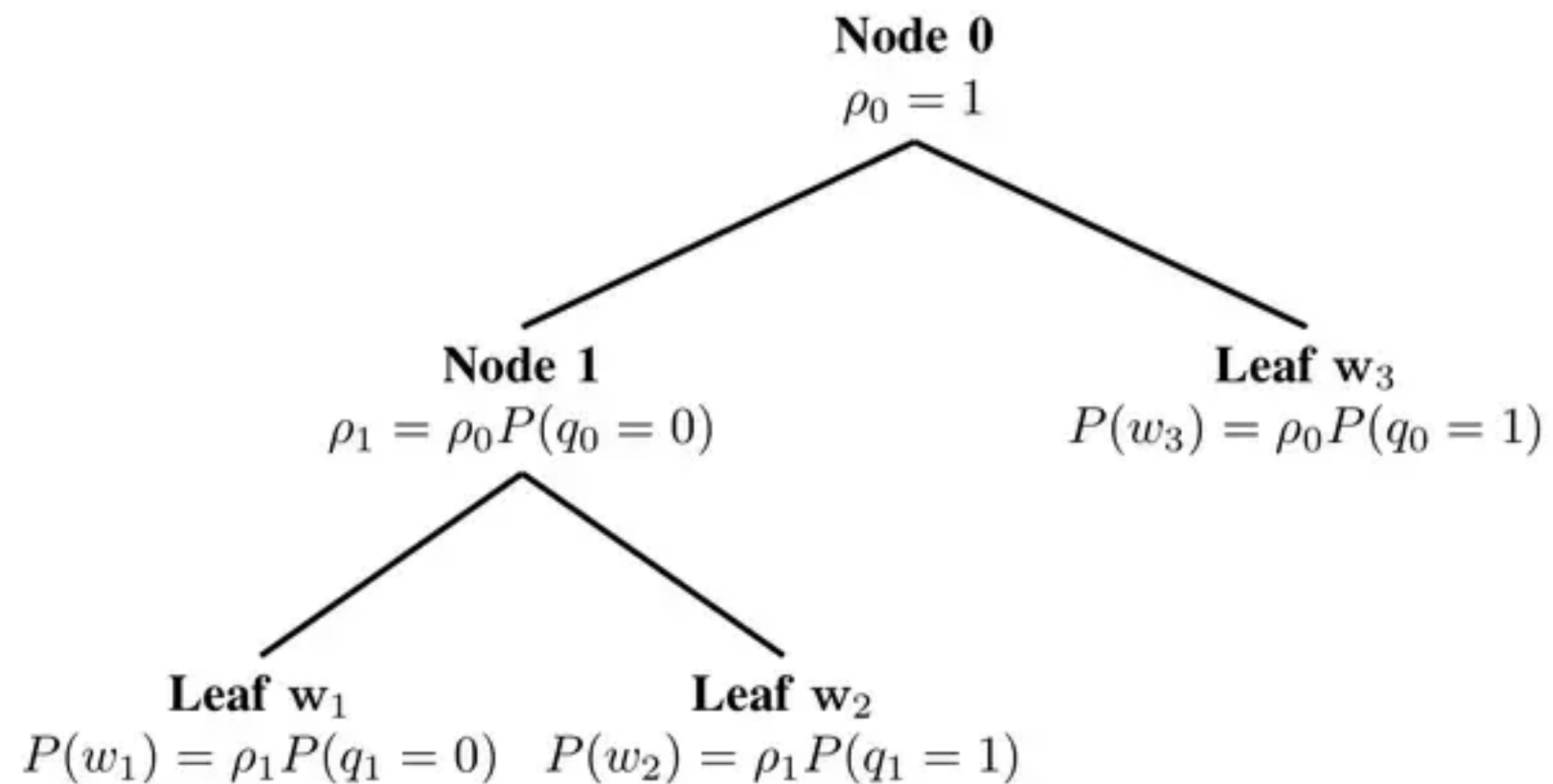
Other ways to sample:

Importance Sampling (Bengio and Senecal 2003)

Noise Contrastive Estimation (Mnih & Teh 2012)

Hierarchical softmax

(Morin and Bengio 2005)



(figure credit: [Quora](#))

Class based softmax

- ▶ Two-layer: cluster words into **classes**, predict class and then predict word.

$$P(c|h) = \text{softmax}(W_c h + b_c)$$

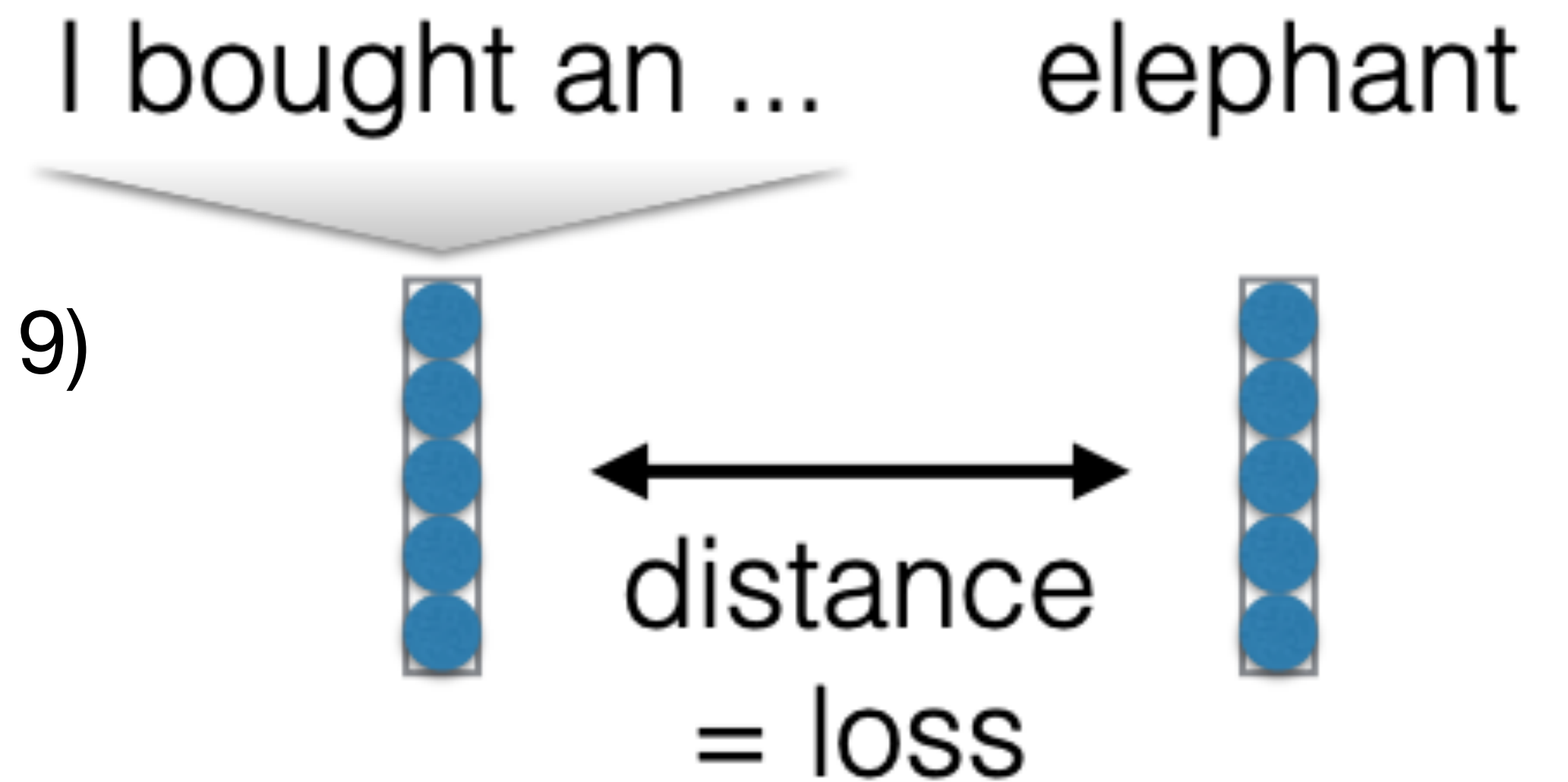

$$P(x|c,h) = \text{softmax}(W_x h + b_x)$$


(figure credit: Graham Neubig)

- ▶ Clusters can be based on *frequency*, *random*, or *word contexts*.

Embedding prediction

- ▶ Directly predict embeddings of outputs themselves
- ▶ What loss to use? (Kumar and Tsvetkov 2019)
 - ▶ L2? Cosine?
 - ▶ Von-Mises Fisher distribution loss, make embeddings close on the unit ball



How to improve generation?

- Improve decoding
- Improve training
- Improve training data

Improving decoding

Challenges in text generation

Repetitiveness

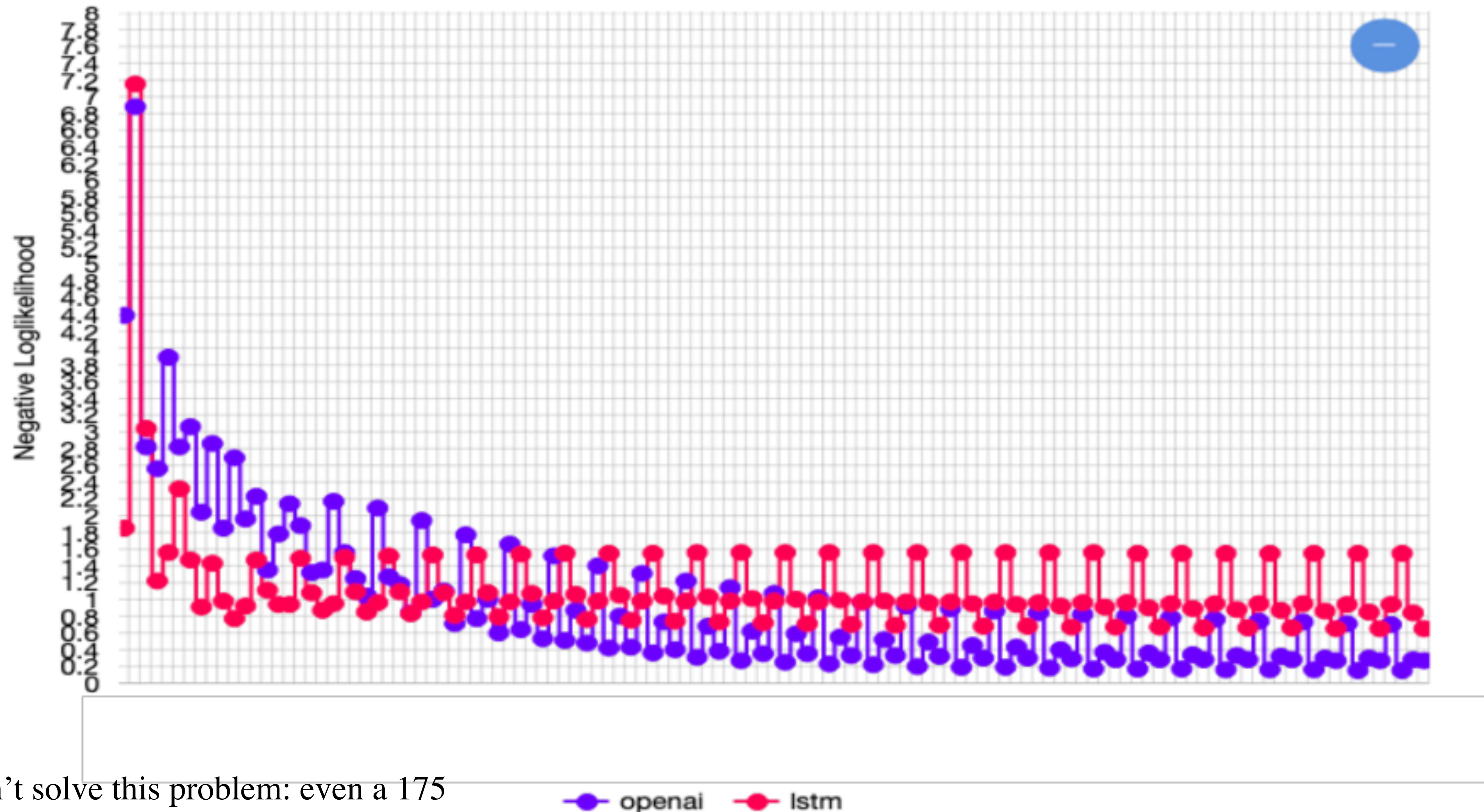
- Diversity
- Exposure Bias
- Evaluation

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.

Continuation: 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...**

Once a model repeats, it keeps repeating

I'm tired. I'm tired. I'm tired. I'm tired. I'm tired. I'm tired. I'm tired. I'm tired. I'm tired. I'm tired.



Scale doesn't solve this problem: even a 175 billion parameter LM still repeats when we decode for the most likely string.

Ways to reduce repetition

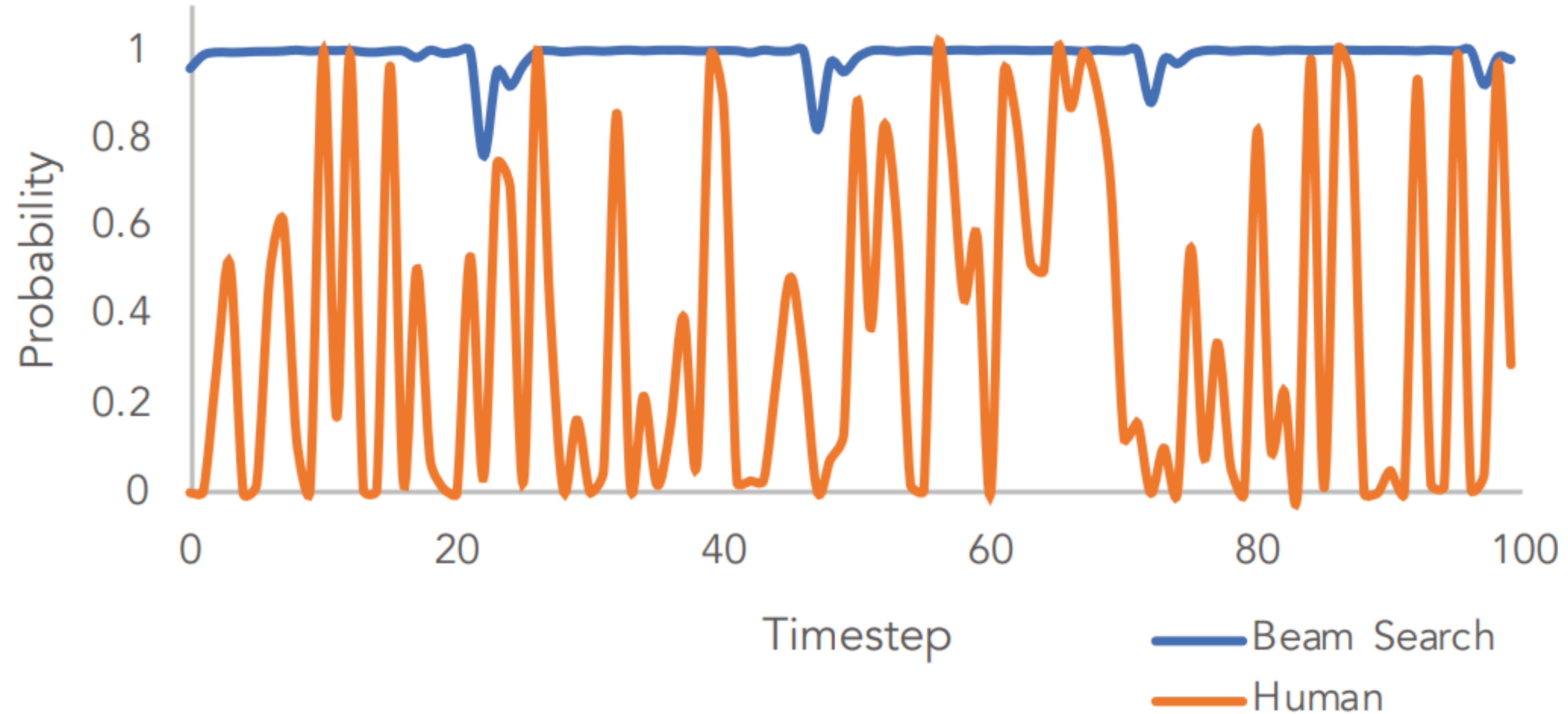
Simple option:

- Heuristic: Don't repeat n -grams

More complex:

- Use a different training objective:
 - Unlikelihood objective (Welleck et al., 2020) penalize 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 x that maximize $\text{logprob_largeLM}(x) - \text{logprob_smallLM}(x)$.

Human generation has lots of diversity!



The Curious Case of Neural Text Degeneration
<https://openreview.net/pdf?id=rygGQyrFvH>
[Holtzman et al, ICLR 2020]

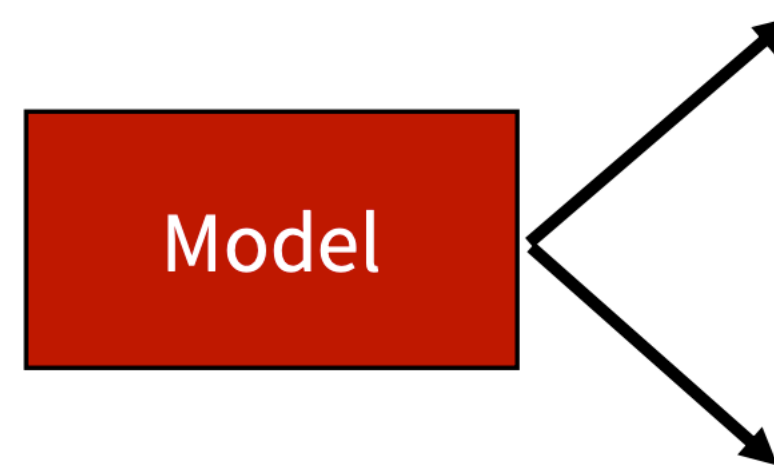
Different ways to sample during decoding

- Basic/vanilla sampling over entire distribution
- Top-k sampling
- Top-p (nucleus) sampling
- Temperature based sampling

Issues with vanilla sampling

- Sample from entire probability distribution

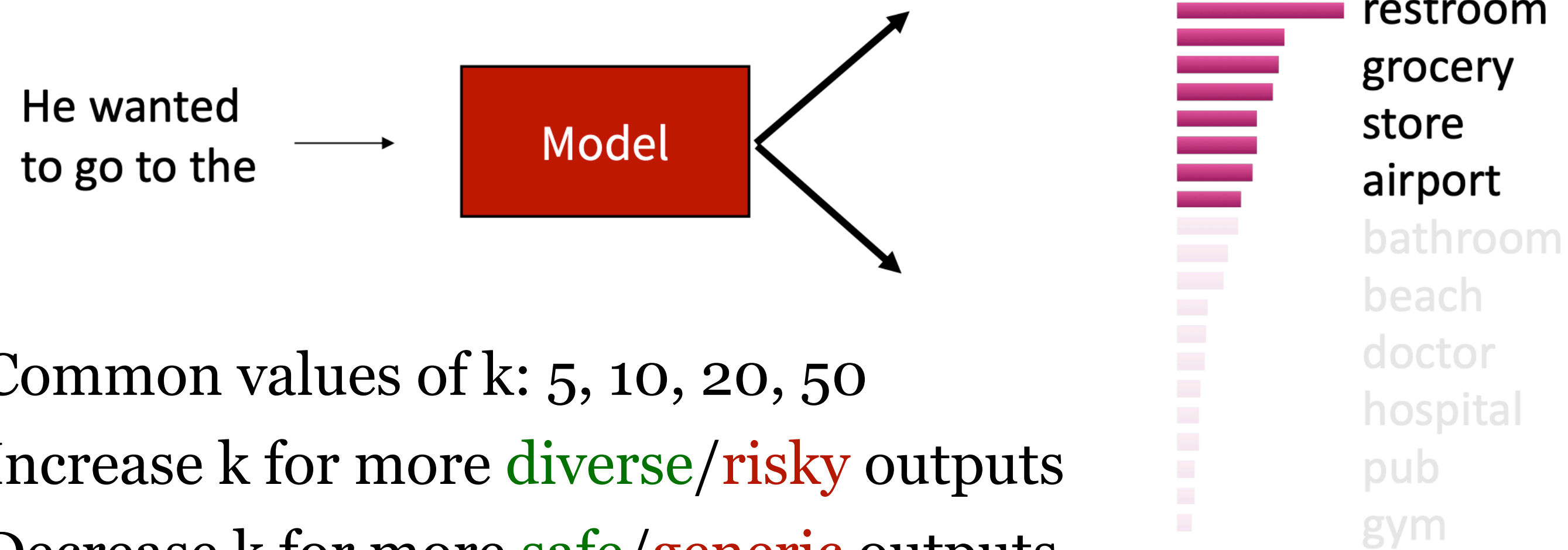
He wanted
to go to the



- Long tail could have enough mass unlikely words are still selected

Decoding: Top-k sampling

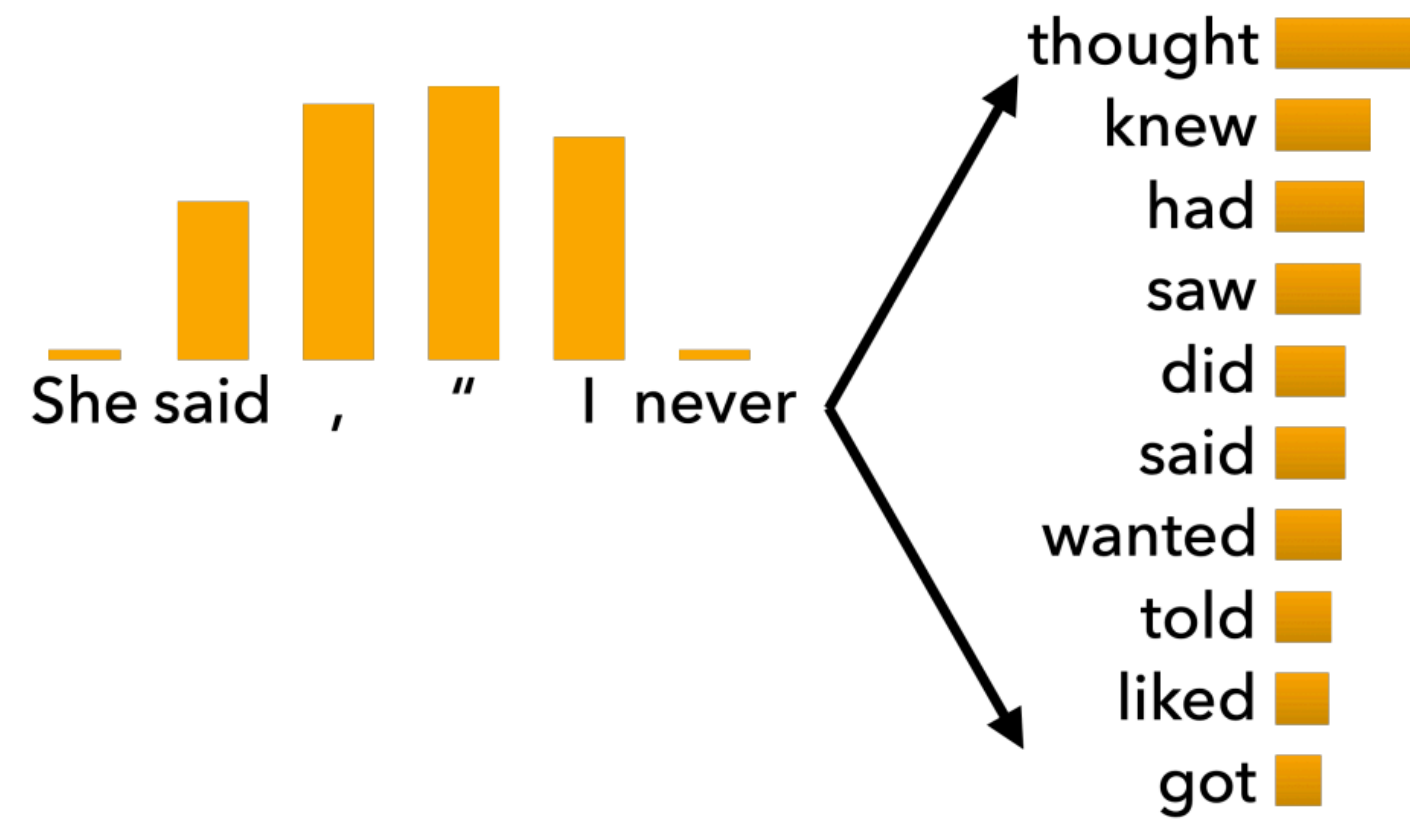
- Only sample from top k tokens in the probability distribution



- Common values of k: 5, 10, 20, 50
 - Increase k for more **diverse/risky** outputs
 - Decrease k for more **safe/generic** outputs
- Greedy search: $k = 1$, Pure sampling: $k = |V|$

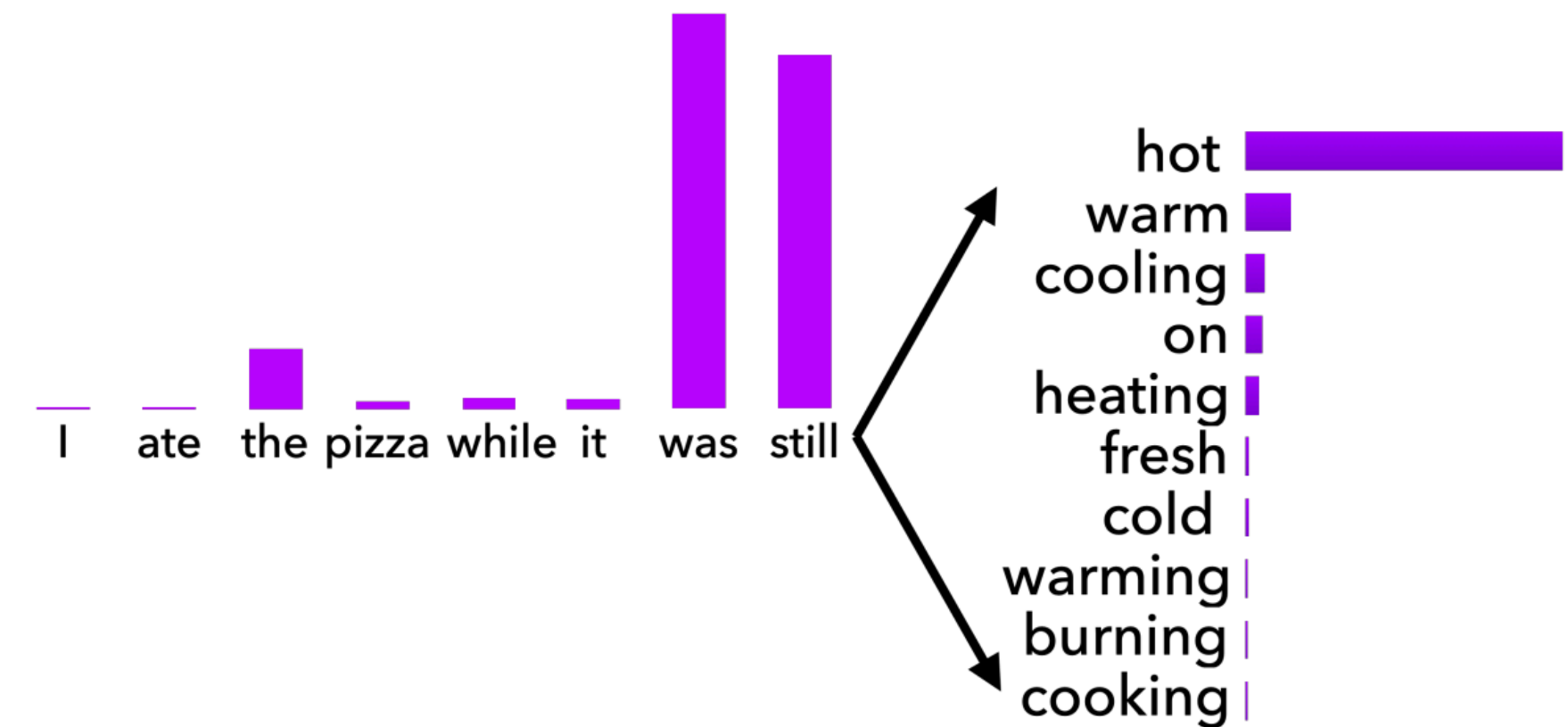
Decoding: Top-k sampling

Cuts off too slowly!



Flat distribution

Cuts off too quickly!



Peaky distribution

The Curious Case of Neural Text Degeneration
<https://openreview.net/pdf?id=rygGQyrFvH>
[Holtzman et al, ICLR 2020]

Slide adapted from Stanford CS224N (Xiang Lisa Li, Antoine Bosselut, Chris Manning)

Decoding: Top-p (nucleus) sampling

- Sample from all tokens in the **top p** cumulative probability mass
- This allows **k to vary** depending on the peakiness of the distribution P_t

$$P_t^1(y_t = w \mid \{y\}_{<t})$$



$$P_t^2(y_t = w \mid \{y\}_{<t})$$



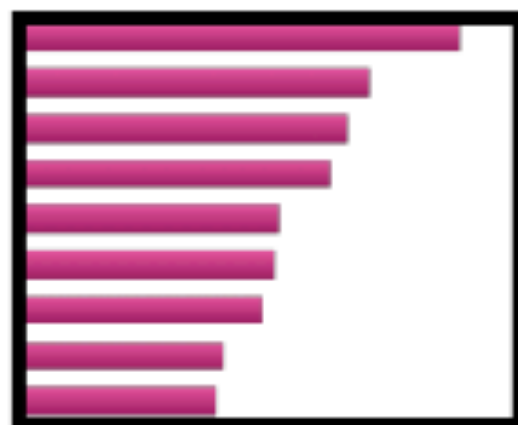
$$P_t^3(y_t = w \mid \{y\}_{<t})$$



Decoding: Other variants

- Typical Sampling [Meister et al. 2022]
 - Reweights the score based on the entropy of the distribution
- Epsilon Sampling [Hewitt et al. 2022]
 - Set threshold for lower bounding valid probabilities

$$P_t^1(y_t = w | \{y\}_{<t})$$



$$P_t^2(y_t = w | \{y\}_{<t})$$



$$P_t^3(y_t = w | \{y\}_{<t})$$



Improving decoding: Temperature scaled softmax

- Recall: On timestep t , the model samples from the distribution P_t which is computed by taking the softmax of a vector of scores $S \in \mathbb{R}^{|V|}$

$$P_t(y_t = w) = \frac{\exp(S_w)}{\sum_{w' \in V} \exp(S_{w'})}$$

- We can apply a **temperature hyperparameter** τ to the softmax to rebalance the distribution

$$P_t(y_t = w) = \frac{\exp(S_w / \tau)}{\sum_{w' \in V} \exp(S_{w'} / \tau)}$$

- Raise the temperature $\tau > 1$: P_t becomes more uniform
 - More diverse output (probability is spread around vocabulary)
- Lower the temperature $\tau < 1$: P_t becomes more spiky
 - Less diverse output (probability is concentrated on top words)

Note: temperature scaled softmax **is not a decoding algorithm!**
It's a **decoding hyperparameter** you can apply at test time, in conjunction with a decoding algorithm (such as beam search or sampling)

Improving Decoding: Re-ranking

- Decode a bunch of sequences (say 10) and **re-rank** with a score that measure the quality of the sequences
- Have a separate scoring function to approximate the quality of the sequences
 - Simplest is to use **low perplexity**
 - But repetitive sequences can have low perplexity...
 - Re-rankers can score a **variety of properties**
 - style, discourse, logical consistency, factuality, etc
 - Can combine these different rankers (but beware of poorly-calibrated re-rankers)

Improving training

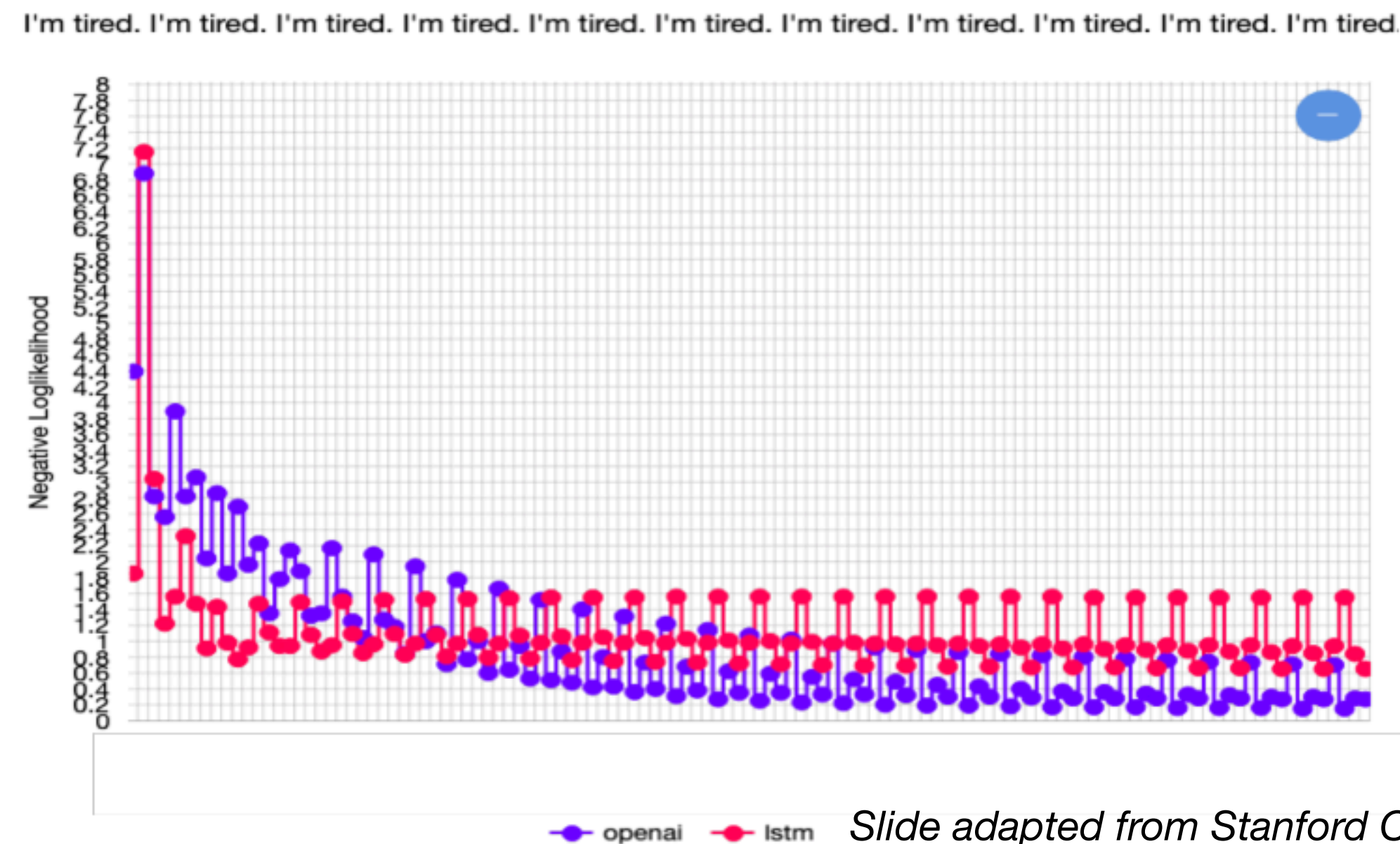
Training the model

- Maximum Likelihood Training: trained to minimize the negative log-likelihood of the next token y_t^* given the preceding tokens in the sequence $\{y^*\}_{<t}$

$$\mathcal{L} = - \sum_{t=1}^T \log P(y_t^* | \{y_{<t}^*\})$$

- Maximum Likelihood Training discourages diverse text generation.

Also known as
teacher forcing



Slide adapted from Stanford CS224N (Xiang Lisa Li, Antoine Bosselut, Chris Manning)

Reducing exposure bias

- **Scheduled sampling** (Bengio et al., 2015)
 - With some probability p , **decode a token** and feed that as the next input, rather than the **gold token**.
 - Increase p over the course of training
 - Leads to improvements in practice, but can lead to **strange training objectives**
- **Dataset Aggregation** (DAgger; Ross et al., 2011)
 - At various intervals during training, generate sequences from your current model
 - **Add these sequences** to your training set as additional examples

Unlikelihood Training

- Given a set of undesired tokens \mathcal{C} , **lower their likelihood in context**

$$\mathcal{L}_{UL}^t = - \sum_{y_{neg} \in \mathcal{C}} \log(1 - P(y_{neg} \mid \{\mathbf{y}^*\}_{<t}))$$

- Keep **teacher forcing** objective and **combine them** for final loss function

$$\mathcal{L}_{MLE}^t = -\log P(y_t^* \mid \{\mathbf{y}^*\}_{<t})$$

$$\mathcal{L}_{ULE}^t = \mathcal{L}_{MLE}^t + \alpha \mathcal{L}_{UL}^t$$

- Set $\mathcal{C} = \{\mathbf{y}^*\}_{<t}$ and you'll train the model to lower the likelihood of previously-seen tokens!
 - Limits repetition!
 - Increases the diversity of the text you learn to generate!

Neural text generation with unlikelihood training

<https://openreview.net/pdf?id=SJeYe0NtvH>

[Welleck et al, ICLR 2020]

Slide adapted from Stanford CS224N (Xiang Lisa Li, Antoine Bosselut, Chris Manning)

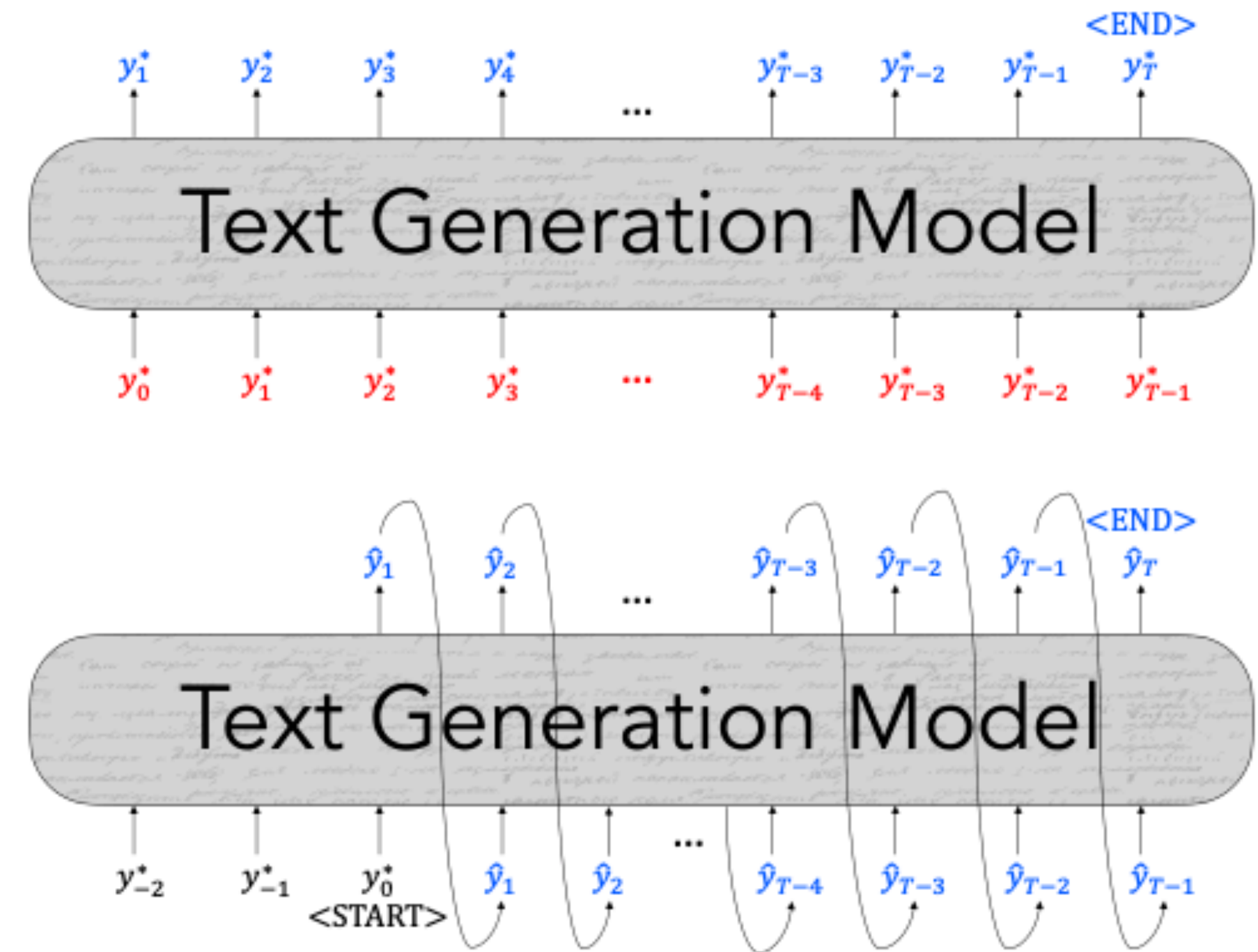
Exposure bias

- Discrepancy in model input between training and generation time
- During training, model inputs are gold context tokens

$$\mathcal{L}_{MLE} = - \sum_{t=1}^T \log P(y_t^* | \{y_{<t}^*\})$$

- At generation time, inputs are previously-decoded tokens

$$\mathcal{L}_{dec} = - \sum_{t=1}^T \log P(\hat{y}_t | \{\hat{y}_{<t}\})$$



Student forcing: use predicted tokens during training

Scheduled sampling: use decoded token with some probability p, increase p over time

Use reinforcement learning

- Sample sequence from your model
- Increase probability of sampled token in the same context
 - Proportional to reward based on reward function

$$\mathcal{L}_{RL} = - \sum_{t=1}^T r(\hat{y}_t) \log P(\hat{y}_t | \{y^*\}; \{\hat{y}_{<t}\})$$

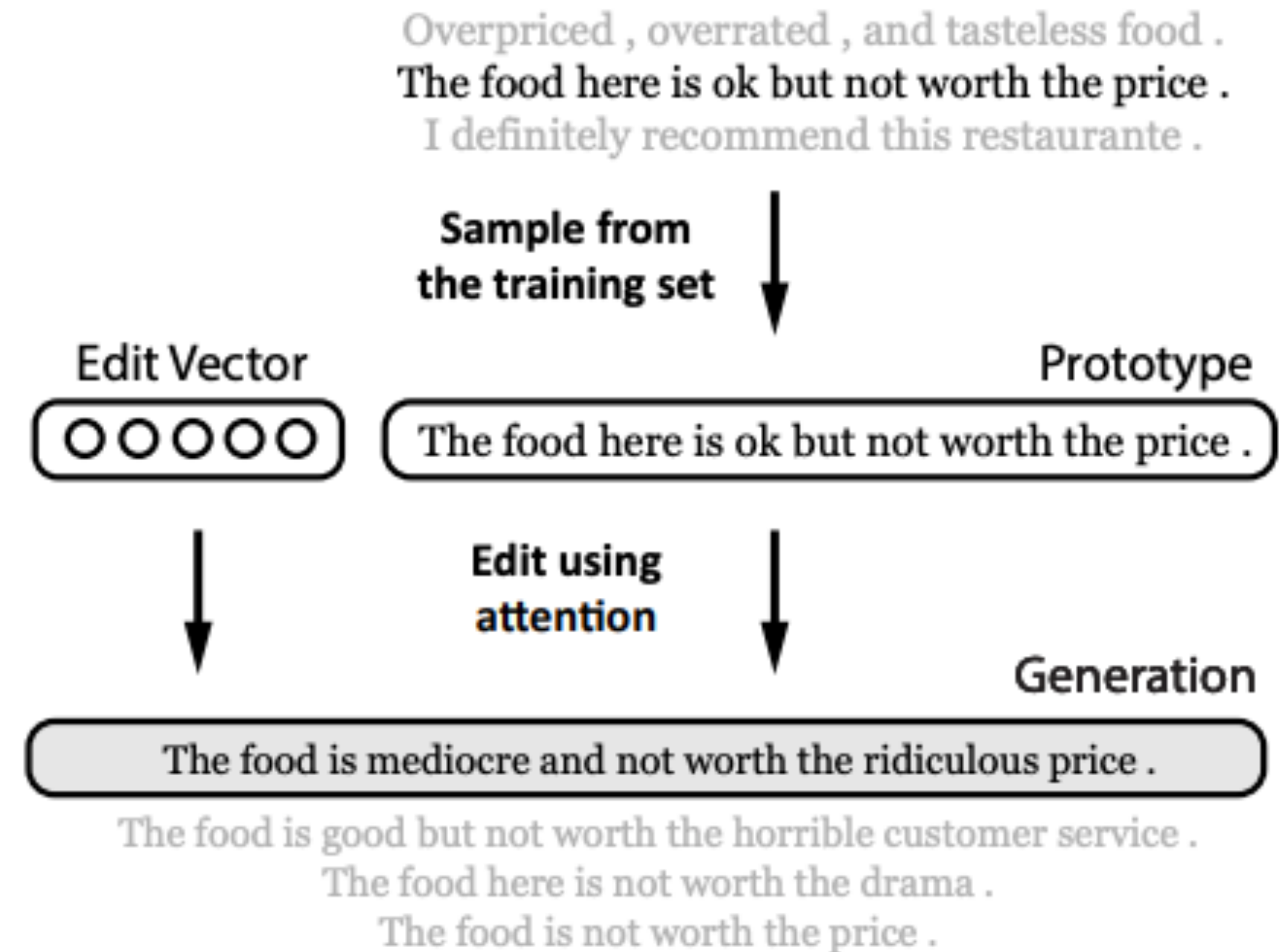
- Reward is based on evaluation metric (BLEU, ROUGE, etc)
- Be careful about “gaming” the reward
 - Your metric will go up! But will your generation actually be better?

“even though RL refinement can achieve better BLEU scores, it barely improves the human impression of the translation quality” – Wu et al., 2016

Alternatives to autoregressive generation

Retrieve and Edit

- Retrieve prototype sentence x' from a corpus
- Sample edit vector z (encodes type of edit to be perform).
- Use neural editor to combine edit vector z and prototype sentence x' to get new sentence x .



Non-autoregressive generation (with transformers)

- Can generate words in a non-autoregressive manner
- Relies on the idea of masked language model
- Predict length of output
- Iterative refinements / masking
 - Predict length of output
 - Predict all words $P(y_i|x)$ $\hat{y}_t^0 = \arg \max_{y_t} \log p(y_t^0|X)$
 - Iteratively refine sequence of predictions based on input and previous predictions
- Efficient decoding since parts of the decoding can run in parallel

$$\hat{y}_t^l = \arg \max_{y_t} \log p(y_t^l | \hat{Y}^{l-1}, X)$$

Each iteration, can just mask out low-confidence words


Mask-Predict: Parallel Decoding of Conditional Masked Language Models

<https://arxiv.org/pdf/1904.09324.pdf>

[Ghazvininejad et al, EMNLP 2019]

Evaluation

Ref: They walked **to the** grocery **store** .
Gen: **The woman** went **to the** **hardware** store .



Content Overlap Metrics



Model-based Metrics



Human Evaluations

Content overlap metrics

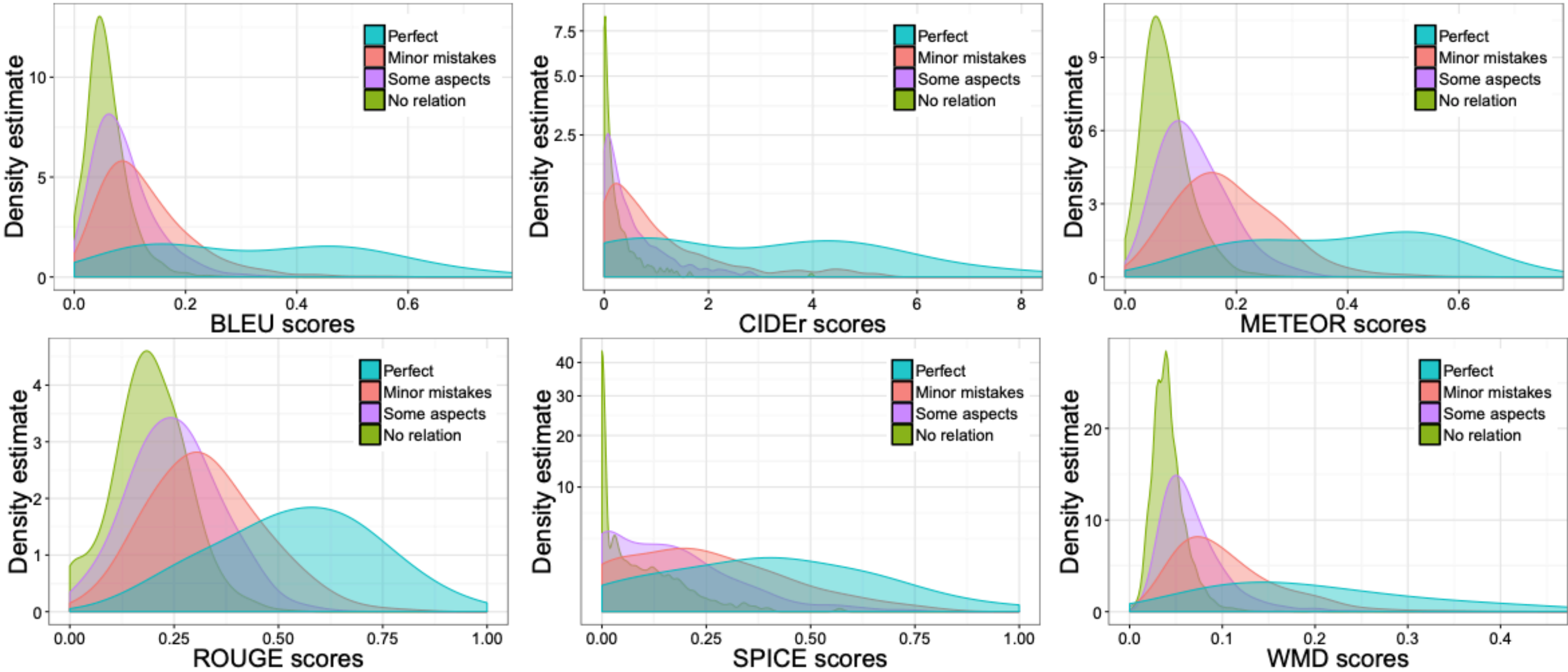
Ref: They walked **to the** grocery **store** .

Gen: **The woman went** **to the** **hardware** **store** .



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

	FLICKR-8K			COMPOSITE		
	Pearson	Spearman	Kendall	Pearson	Spearman	Kendall
WMD	0.68	0.60	0.48	0.43	0.43	0.32
SPICE	0.69	0.64	0.56	0.40	0.42	0.34
CIDEr	0.60	0.56	0.45	0.32	0.42	0.32
METEOR	0.69	0.58	0.47	0.37	0.44	0.33
BLEU	0.59	0.44	0.35	0.34	0.38	0.28
ROUGE	0.57	0.44	0.35	0.40	0.39	0.29



N-gram overlaps are not good metrics

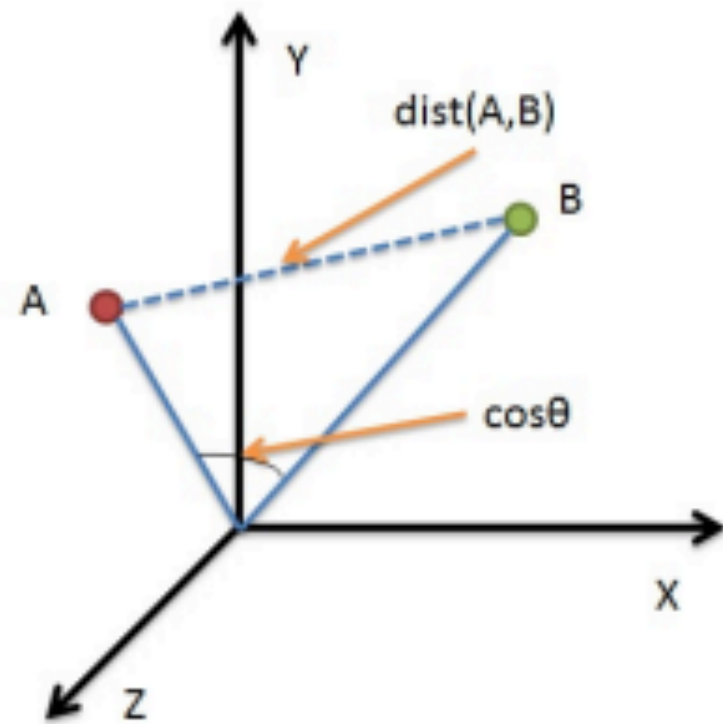
Word overlap-based metrics: BLEU, ROUGE, METEOR, CIDEr, etc

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

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!
- Even though embeddings are pretrained, distance metrics used to measure the similarity can be fixed

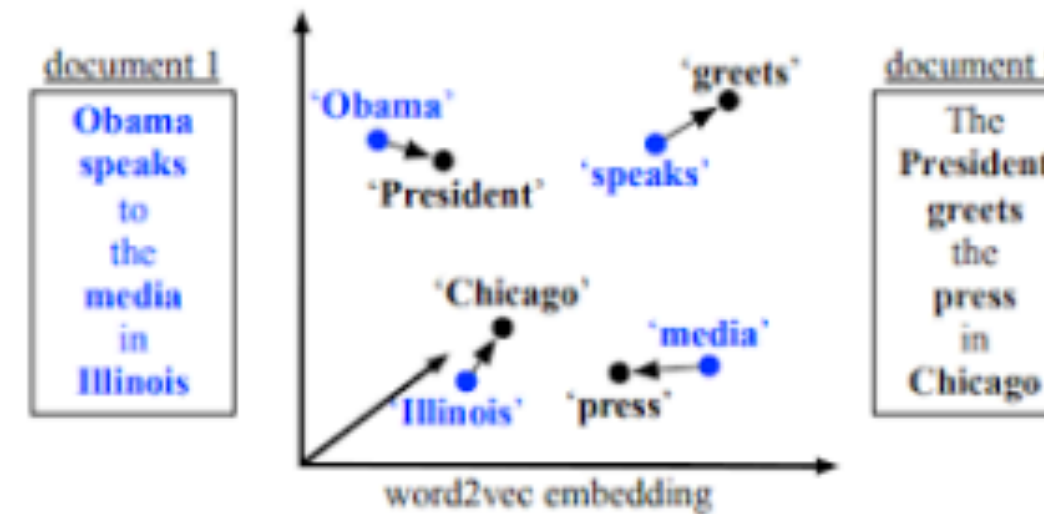
Model-based metrics: Word distance functions



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:

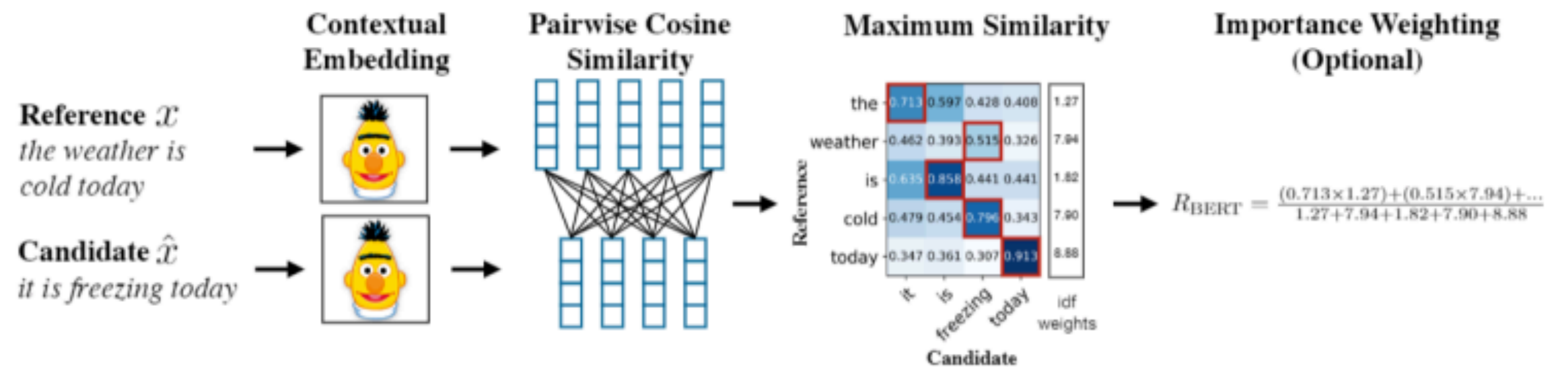
Measures the distance between two sequences (e.g., sentences, paragraphs, etc.), using word embedding similarity matching.

(Kusner et.al., 2015; Zhao et al., 2019)

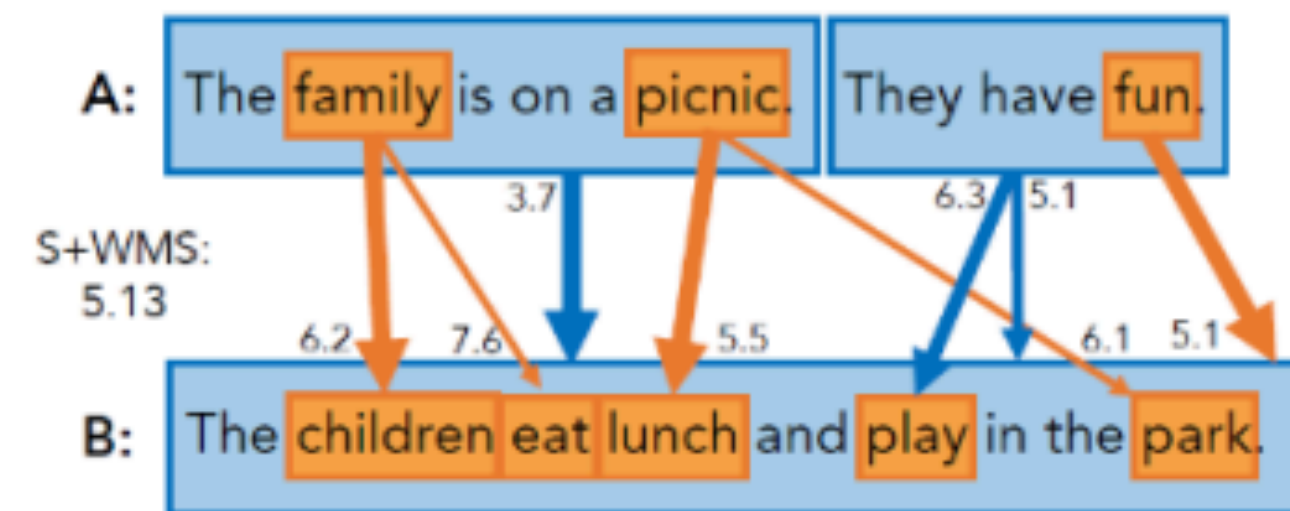
BERTSCORE:

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

(Zhang et.al. 2020)



Model-based metrics: Beyond word matching



Sentence Movers Similarity :

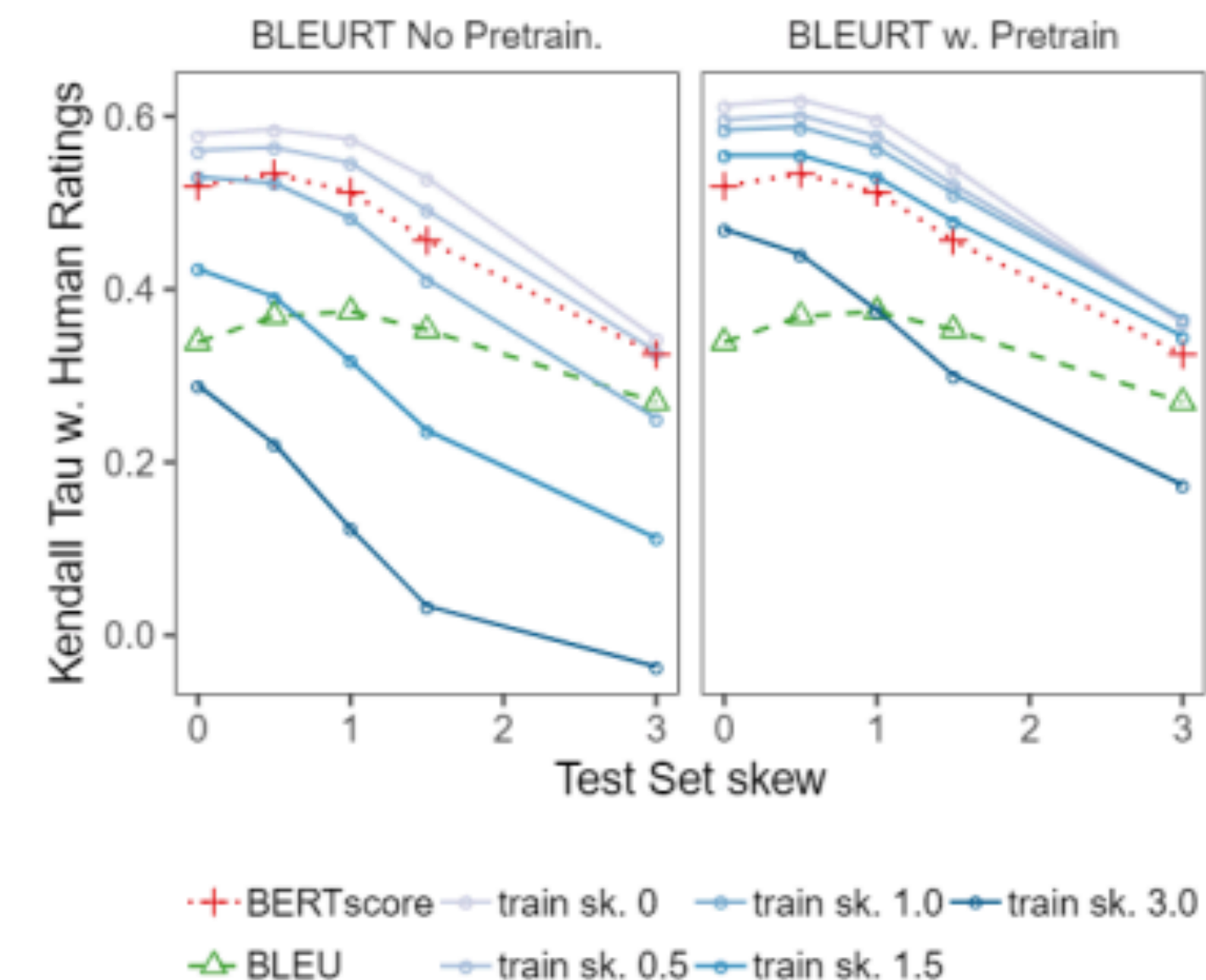
Based on Word Movers Distance to evaluate text in a continuous space using sentence embeddings from recurrent neural network representations.

(Clark et.al., 2019)

BLEURT:

A regression model based on BERT returns a score that indicates to what extent the candidate text is grammatical and conveys the meaning of the reference text.

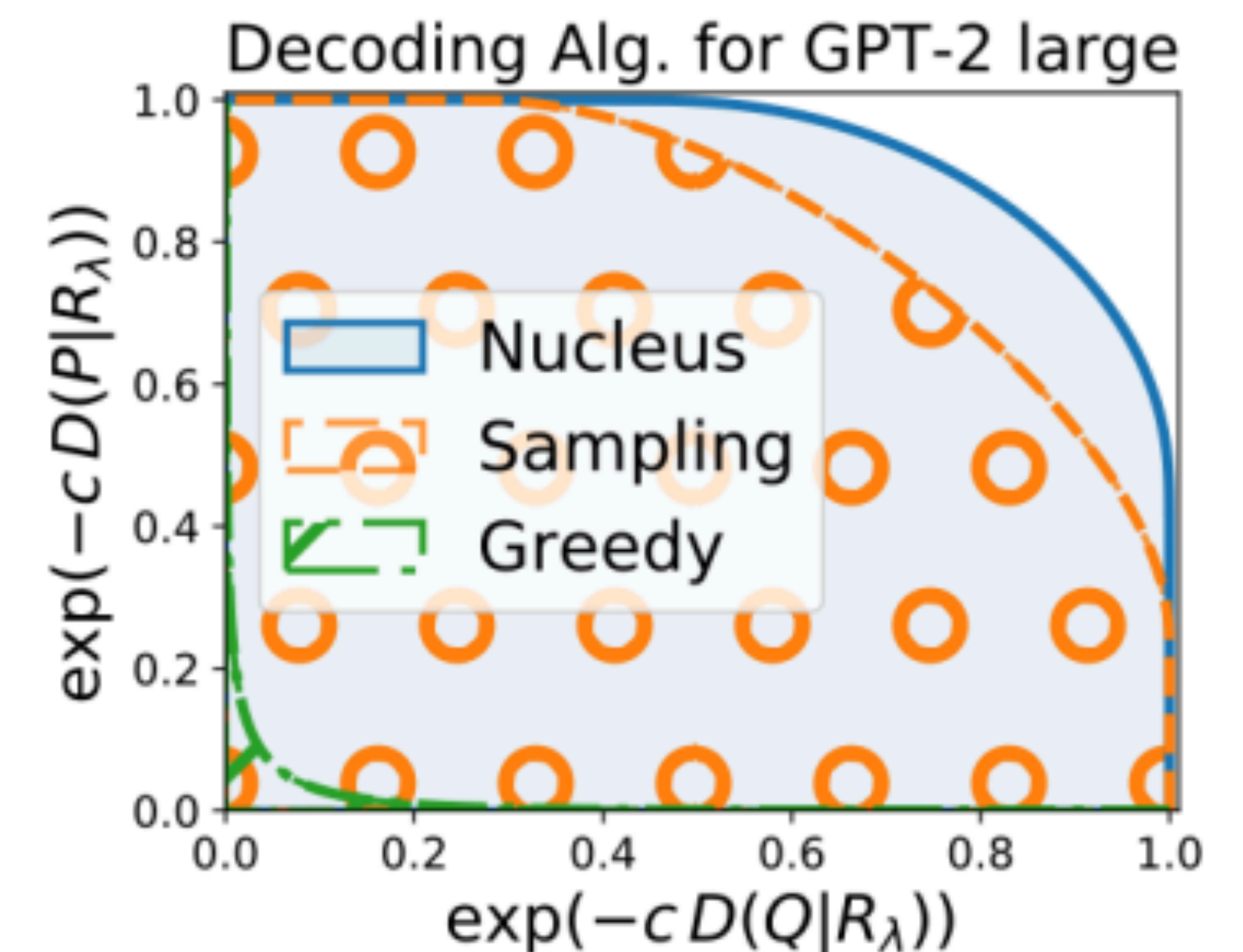
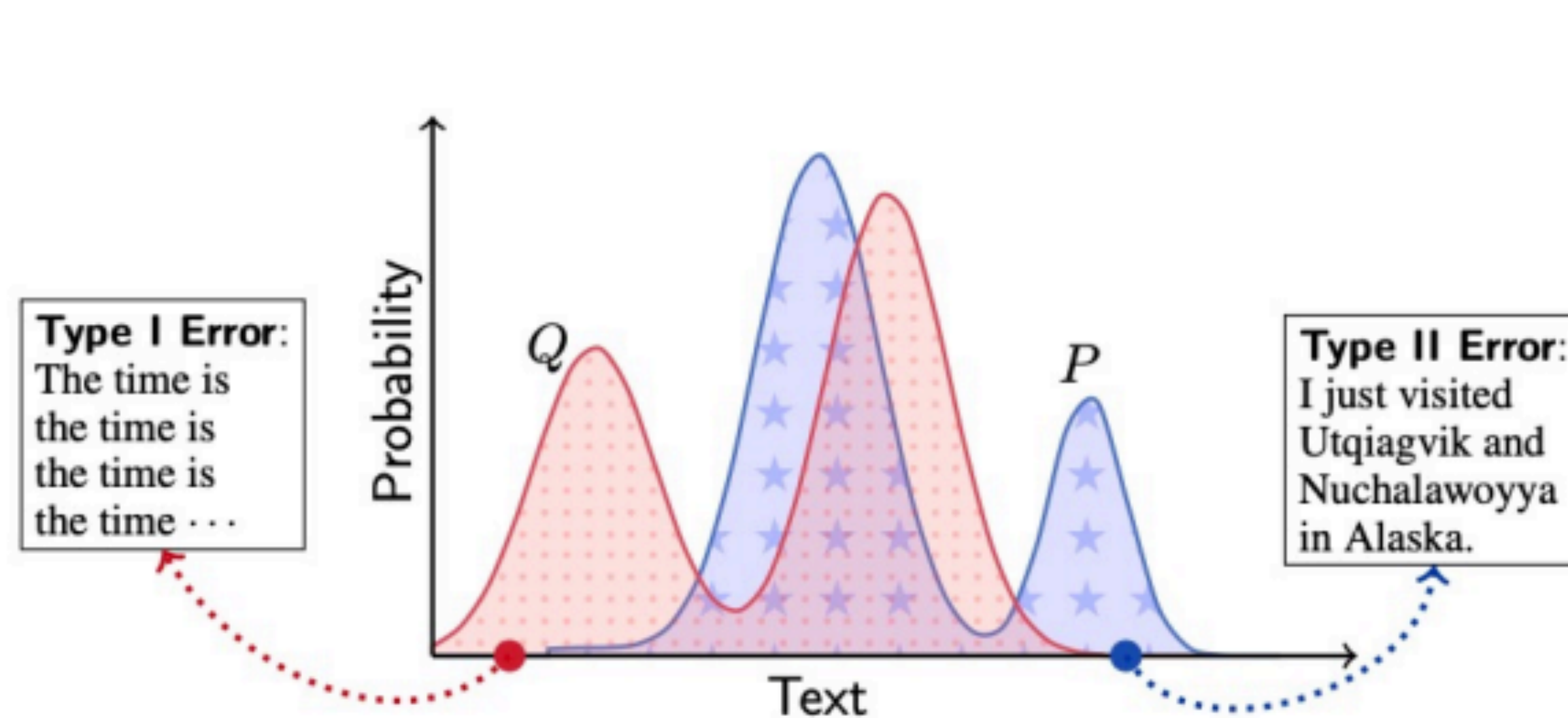
(Sellam et.al. 2020)



Evaluating open-ended text generation

MAUVE

MAUVE computes information divergence in a quantized embedding space, between the generated text and the gold reference text (Pillutla et.al., 2022).

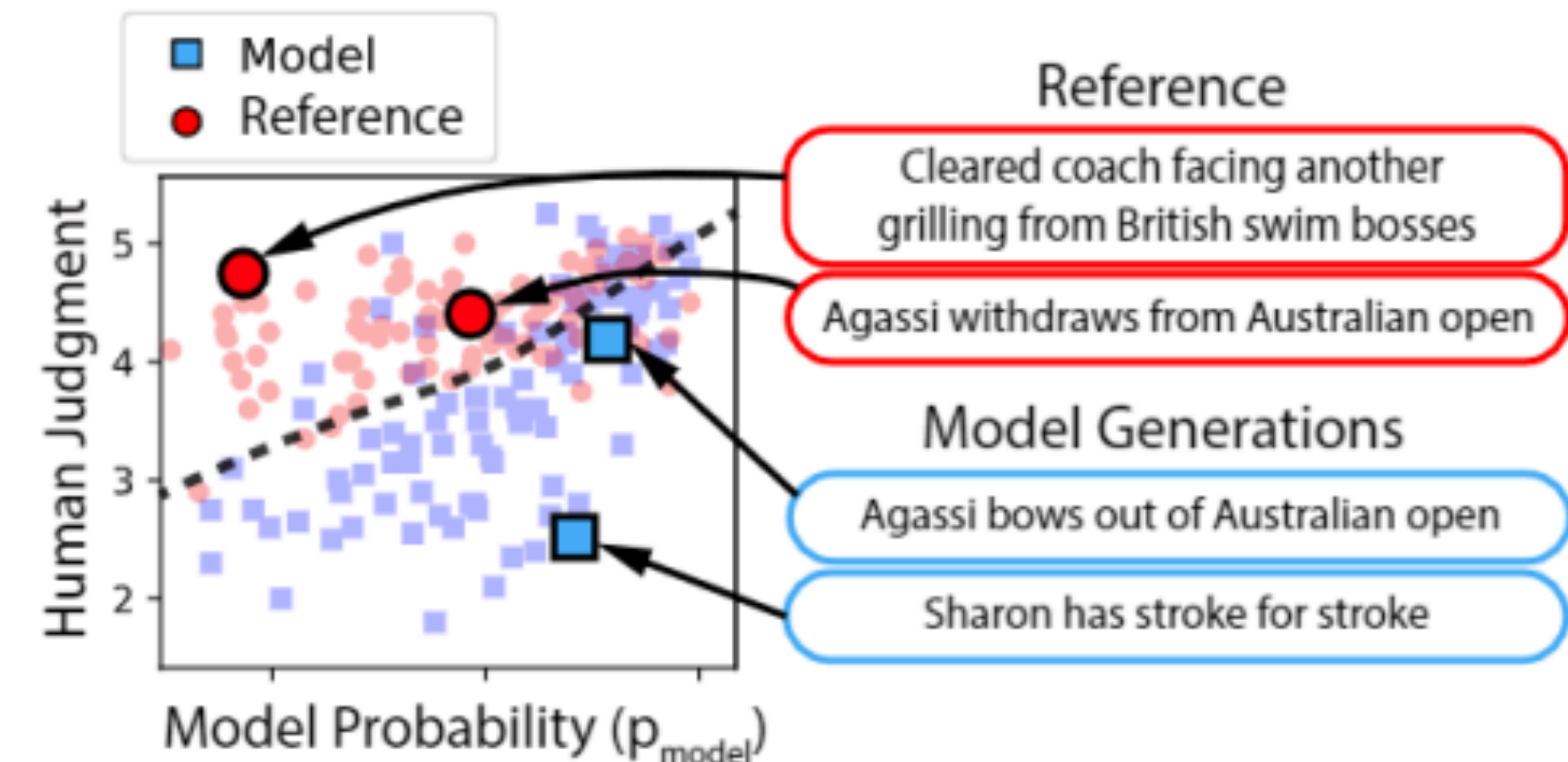
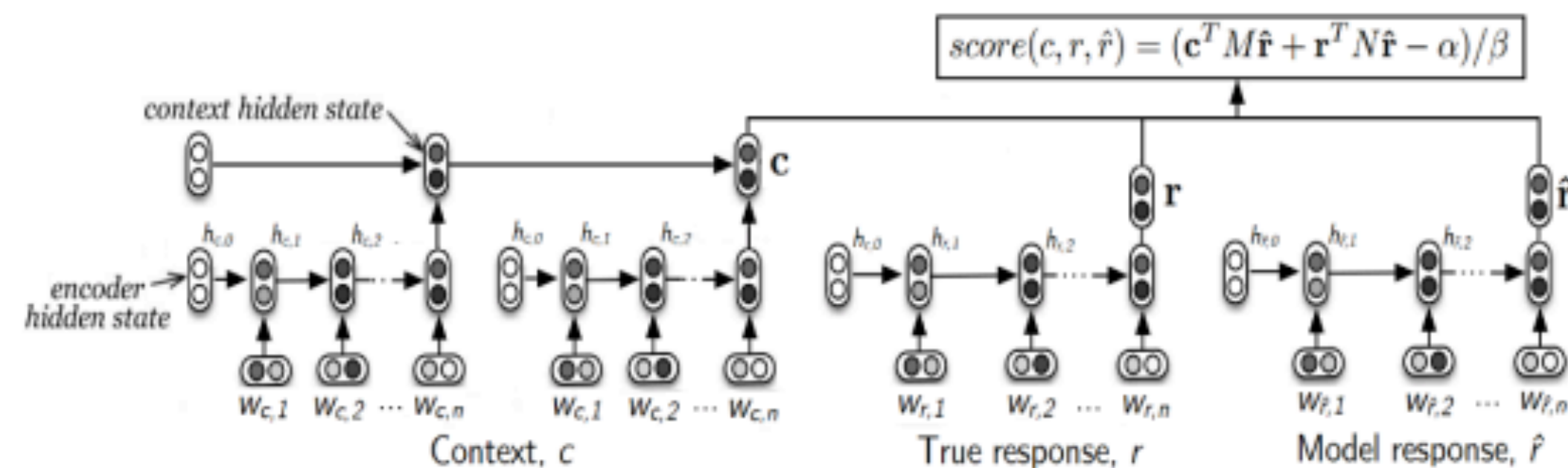


Human evaluation



- [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
- Drawbacks
 - Cannot compare across studies
 - Inconsistent
 - Slow and expensive
 - Does not measure recall
- When developing new automatic metrics, human evaluation is used as gold.
 - New automated metrics must correlate well with human evaluation.

Learning from human feedback



ADEM:

A learned metric from human judgments for dialog system evaluation in a chatbot setting.

(Lowe et.al., 2017)

HUSE:

Human Unified with Statistical Evaluation (HUSE), determines the similarity of the output distribution and a human reference distribution.

(Hashimoto et.al. 2019)

Evaluation takeaways

- **Content overlap metrics** provide a good starting point for evaluating the quality of generated text, but they're not good enough on their own.
- **Model-based metrics** can be more correlated with human judgment, but metric may not be not interpretable
- **Human judgments** are critical
 - But humans are **inconsistent** and judgments are **expensive**
- If you are developing a NLG system, you should
 - Look at your model generations. Don't just rely on numbers!
 - Publicly release large samples of the output of systems that you create!