UNIVERSITY OF CAMBRIDGE

Language Model Decoding Beyond Beam Search:

Recent decoding methods and why you should use them

Julius Cheng

Department of Computer Science and Technology, University of Cambridge, UK

4 September 2024

Goals of this talk

1. To cover recent methods for MT decoding.
2. To help practitioners navigate the vast & varied literature on decoding to find the right tools for their use case.
3. To provide interested researchers with insight into why these methods work, and where they can be improved upon.

Scope of this talk

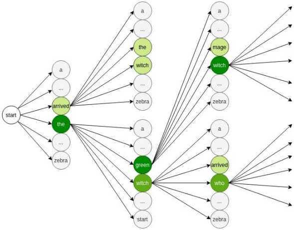
- Assume autoregressive LMs trained in the usual ways.
- No LM training-time methods (RL, fine-tuning).

Language model (LM) decoding

- LMs for NMT model $p(y_t|x, y_{t-1})$, which induces sequence distribution $p(y|x)$.
- Output sequence y or sequences $\mathbf{y}$ which optimize some criterion.
- Criteria:
  - Output y with the highest human-rated translation quality ("prediction").
  - Output $\mathbf{y}$ which maximizes a combination of human-rated quality and lexical diversity ("diverse decoding").

Baseline methods for prediction

- Beam search
  - At time t, keep k candidates $\mathbf{c}_t$ of length t.
  - To get $\mathbf{c}_{t+1}$, take the k highest-probability continuations of $\mathbf{c}_t$ of length $t+1$.
  - Greedy search is beam search with $k = 1$.



https://www.baeldung.com/cs/beam-search

Baseline methods for diverse decoding

- Ancestral sampling
  - Draw next-token y_{t+1} from $p(y_{t+1}|x, y_t)$, stop when EOS is reached.
  - Optional: mitigate low quality samples by warping the next token distribution.
    - Temperature scaling: rescale logits z to z/τ before softmax
    - Truncation methods: set certain token probabilities to 0, then renormalize, e.g. top-k, nucleus sampling.

MT Prediction

The goal of prediction is return y which minimizes loss (error). Loss can be with measured with respect to the source x, reference y^*, or both.

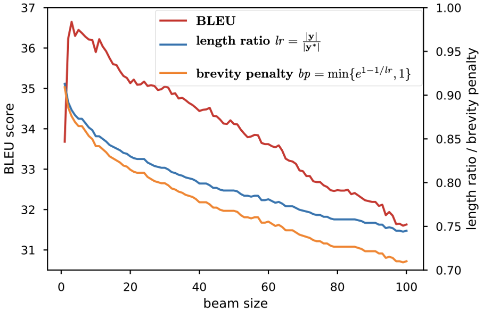
- $\mathcal{L}(y, y^*)$ - most automatic metrics, e.g. BLEU, ROGUE, METEOR, chrF++, BLEURT
- $\mathcal{L}(x, y)$ - "quality estimation", e.g. OpenKiwi, referenceless COMET
- $\mathcal{L}(x, y, y^*)$ - e.g. reference-based COMET

But these are only proxies to the true loss: human ratings.

Why is beam search suboptimal?

Beam search curse:

- Maximizing probability hurts beyond a point ("inadequacy of the mode")



Yang et al., 2018

Why is beam search suboptimal?

Beam search curse:

- The true distribution mode is the empty sequence in as much as 50% of translations (Stahlberg and Byrne, 2019)

Why is beam search suboptimal?

Beam search curse:

- The true distribution mode is the empty sequence in as much as 50% of translations (Stahlberg and Byrne, 2019)
- Relationship between *intrinsic uncertainty* of a task and the adequacy of the mode - no beam search curse for grammatical error correction (GEC). (Stahlberg and Byrne, 2022).

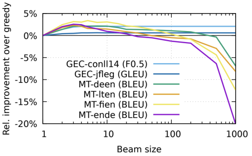


Figure 2: Relative beam search improvements over greedy search. MT quality degrades with large beam sizes, but GEC saturates after a beam size of 10.

Reranking

If sequence probability is flawed, rerank with better-aligned criterion.

- *n*-best reranking
  1. Obtain candidates $\mathbf{y}$ (e.g. with beam top-k, sampling)
  2. Obtain reranked scores $s(y), y \in \mathbf{y}$, for some scoring function s.
  3. Return $\arg\max_{y \in \mathbf{y}} s(y)$.

Quality-based reranking

- Set s to a referenceless quality estimator.
- The better quality estimators rely on human annotations. What if you don't have them?

Discriminative rerankers

Discriminative Reranking for NMT (Lee et al., 2021)

- Train a bidirectional encoder to output $s(x, y)$.
- The reranked probability of a candidate is $p_M(y|x) = \frac{exp(s(y,x))}{\sum_{y' \in \mathbf{y}} \exp(s(y',x))}$ given an n-best list $\mathbf{y}$.
- Trained to match the distribution $p_T(y|x) = \frac{\exp(\mu(y,r))/\tau}{\sum_{y' \in \mathbf{y}} exp(\mu(y',r))/\tau}$
  - μ: the desired metric (BLEU)
  - r: the true reference
  - τ: temperature

Discriminative rerankers

Energy-Based Reranking (Bhattacharyya et al., 2021)

- Same joint-encoder architecture as previous, except energy E is defined as the average of per-token scalar values.
- Trained with a **margin**-based loss.
- Margin violation: negative difference in energy between two candidates must be at least as large as their difference in BLEU.
- Interpolate LM and reranker scores:
  $p(y|x) \propto p_\theta(y|x) \exp(-E(y,x)/\tau)$

Discriminative rerankers

Discriminative reranking & energy-based reranking

- Both use a joint-encoder transformer
- Both use BLEU scores of random samples against a gold reference
- Mainly differ in a KL vs. margin objective

Noisy channel decoding

Noisy channel decoding (Yee et al., 2019)

- Using Bayes rule, $p(y|x) \propto p(x|y)p(y)$.
- In practice, actually use a mixture $\log p(y|x) + \lambda(\log p(x|y) + \log p(y))$ for mixture weight λ.
- Why?
  - Modeling $p(y_t|x, y_1, ..., y_t)$ directly can fail with highly predictive prefixes $y_1, ..., y_t$, causing detachment from source (hallucination).
  - "Ensembles" models with different advantages (bidirectional source encoder, unidirectional target decoder).
  - Language model $p(y)$ can be trained on large monolingual corpora.

Noisy channel decoding

Two algorithms presented:

- **Incremental decoding**: Beam search-like algorithm which rescores with the noisy channel mixture.
  - This requires using the reverse model on partial targets, e.g. $p(x|y_1, ..., y_t)$ (which it isn't trained on!), but works okay.

Noisy channel decoding

Two algorithms presented:

- ***n*-best list reranking**

	5	10	50	100
DIR	39.1	39.2	39.3	39.2
DIR ENS	40.1	40.2	40.3	40.3
DIR+LM	40.0	40.2	40.6	40.7
DIR+RL	39.7	40.1	40.8	40.8
DIR+RL+LM	40.4	40.9	41.6	41.8
CH+DIR	39.7	40.0	40.5	40.5
CH+DIR+LM	40.8	41.5	42.8	43.2

Table 2: Re-ranking BLEU with different n-best list sizes on news2016 of WMT De-En. We compare to decoding with a direct model only (DIR) and decoding with an ensemble of direct models (DIR ENS). Table 5 in the appendix shows standard deviations.

Minimum Bayes risk decoding (MBR)

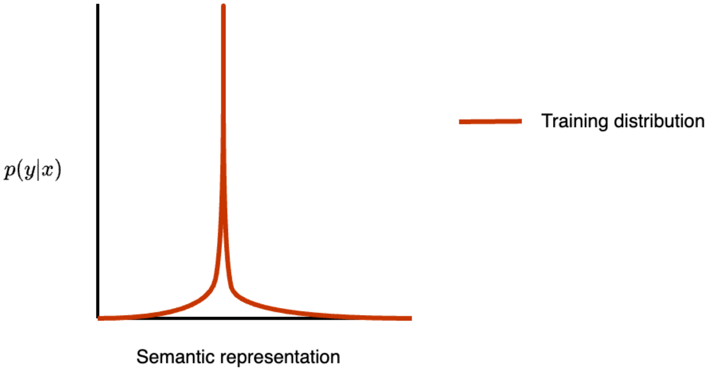
Alternative decoding objective:

$$y^{MBR} = \arg\max_{y} \mathbb{E}_{\hat{y} \sim p_\theta(\cdot|x)} -\mathcal{L}(y, \hat{y})$$

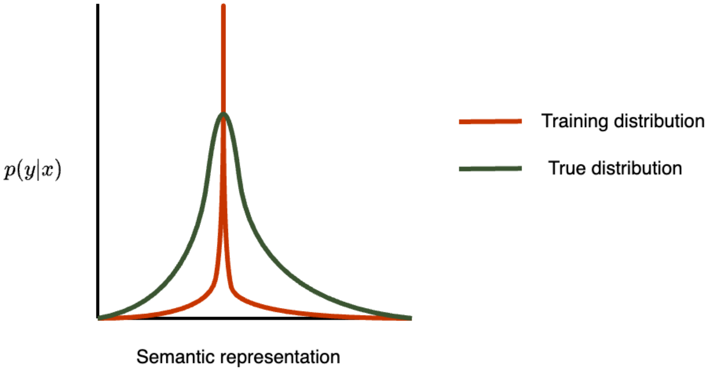
Since $-\mathcal{L}$[1] is a measure of similarity, MBR returns the candidate with the highest *expected similarity* to the model distribution.

[1] Usually called **utility** in the context of MBR

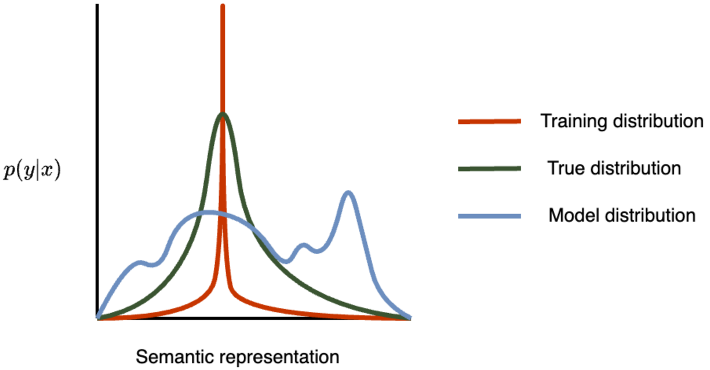
Similarity matters



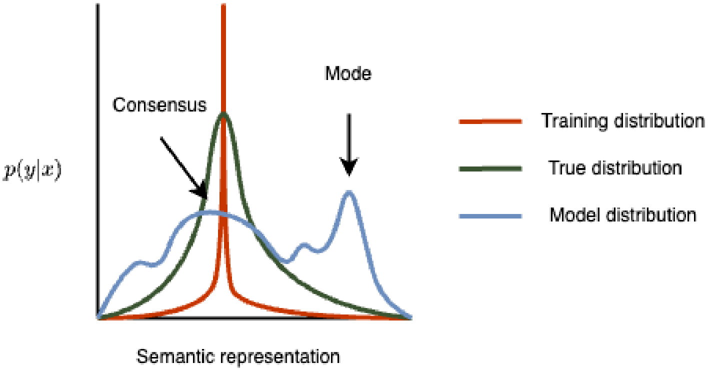
Similarity matters



Similarity matters



Similarity matters



Similarity matters

Mode-seeking is a special case of MBR with a 1-0 exact match loss.

$$\mathcal{L}(y, y') = \begin{cases} 1, & \text{if } y = y' \\ 0, & \text{otherwise} \end{cases}$$

$$y^{MBR} = \arg\max_{y} \mathbb{E}_{\hat{y} \sim p_\theta(\cdot|x)} -\mathcal{L}(y, \hat{y})$$

$$= \arg\max_{y} p(y|x)$$

If MBR is *similarity-sensitive* decoding, then mode-seeking is similarity-*insensitive* decoding.

Similarity matters

I flip a coin 8 times.

You have to guess what sequence comes up.

Should you guess TTTTTTT or THTHHTH?

Minimum Bayes risk decoding (MBR)

I flip a coin 8 times. $\leftarrow$ **language model**

You have to guess what sequence comes up. $\leftarrow$ **decision rule**

Should you guess TTTTTTT or THTHHTH?

If you only win for being right - doesn't matter. If you get partial credit for features, e.g. number of occurences of TH, HH, then guess the latter.

NMT rewards you partial credit, so predict based on **likely features**, not probability!

Similarity-sensitive methods

Similarity-sensitive entropy (Cheng and Vlachos, 2024)

- Common information-theoretic measures for model uncertainty: surprisal/entropy, e.g. average token surprisal (neg. logprob), average entropy: $\sum_{y_t \in \mathcal{V}} p(y_t) \log p(y_t)$.

Similarity-sensitive methods

Similarity-sensitive entropy (Cheng and Vlachos, 2024)

- Common information-theoretic measures for model uncertainty: surprisal/entropy, e.g. average token surprisal (neg. logprob), average entropy: $\sum_{y_t \in \mathcal{V}} p(y_t) \log p(y_t)$.
- We use **similarity-sensitive Shannon entropy** (S3E) to measure *semantic uncertainty* of a distribution:

$$\sum_{y_t \in \mathcal{V}} p(y) \log \mathbb{E}_{y' \sim p(\cdot|x)} \mathcal{S}(y, y')$$

Similarity-sensitive methods

Similarity-sensitive entropy (Cheng and Vlachos, 2024)

- Common information-theoretic measures for model uncertainty: surprisal/entropy, e.g. average token surprisal (neg. logprob), average entropy: $\sum_{y_t \in \mathcal{V}} p(y_t) \log p(y_t)$.
- We use **similarity-sensitive Shannon entropy** (S3E) to measure *semantic uncertainty* of a distribution:

$$\sum_{y_t \in \mathcal{V}} p(y) \log \mathbb{E}_{y' \sim p(\cdot|x)} \mathcal{S}(y, y')$$

- **Similarity-sensitive surprisal** (SSS) of y is the corresponding inner term: $\log \mathbb{E}_{y' \sim p(\cdot|x)} \mathcal{S}(y, y')$.

Similarity-sensitive methods

- Standard entropy may indicate **successful generalization**.
- Standard entropy measures lexical variation, S3E measures *semantic* variation, is more predictive of quality in NMT.

	en-de		et-en		ne-en	
	ρ	r	ρ	r	ρ	r
Prediction-based						
Total token surprisal	0.370	0.205	0.261	0.150	0.402	0.339
Avg. token surprisal	0.352	0.282	0.356	0.333	0.333	0.357
Total token SE	0.218	0.089	0.180	0.078	0.326	0.250
Avg. token SE	0.244	0.251	0.248	0.196	0.242	0.211
SSS, BERT, $\alpha = 0$	0.369	0.344	0.591	0.606	0.573	0.510
SSS, BERT, best α	(5) 0.436	(4) 0.406	(3) 0.648	(4) **0.649**	(6) 0.623	(5) 0.547
Distribution-based						
Sequence SE	0.371	0.232	0.369	0.258	0.567	0.484
Avg. token surprisal	0.315	0.319	0.539	0.542	0.530	0.489
Avg. token SE	0.265	0.280	0.535	0.543	0.545	0.510
S3E, chrF++, $\alpha = 0$	0.138	0.176	0.399	0.417	0.440	0.473
S3E, chrF++, best α	(4) 0.390	(3) 0.332	(3) 0.523	(3) 0.493	(4) 0.591	(4) 0.556
S3E, BERT, $\alpha = 0$	0.304	0.303	0.543	0.568	0.562	0.569
S3E, BERT, best α	(6) **0.487**	(6) **0.424**	(5) **0.655**	(4) 0.647	(6) **0.676**	(5) **0.659**

Similarity-sensitive methods

- Bonus: choose similarity function $\mathcal{S}$ to capture variation over phenonemon of interest.
- Our experiment in named entity token translation - standard methods aren't designed for the task.

	et-en		ne-en	
	ρ	r	ρ	r
Shannon entropy	0.006	0.028	0.158	0.134
Avg. token surprisal	0.025	0.019	0.175	0.209
Avg. token entropy	0.019	0.025	0.196	0.233
S3E, chrF++, $\alpha = 0$	0.172		0.153	0.177
S3E, chrF++, best α	(2) 0.193	(0) 0.243	(3) 0.159	(1) 0.180
S3E, BERT, $\alpha = 0$	0.205	0.287	0.228	0.253
S3E, BERT, best α	(5) 0.239	(2) 0.296	(5) 0.256	(3) 0.274
S3E, NETR, $\alpha = 0$	0.485	0.441		0.346
S3E, NETR, best α	(1) **0.500**	(1) **0.459**	(0) **0.467**	(1) **0.375**

Similarity-sensitive methods

Mode-seeking search $\rightarrow$ MBR

Shannon surprisal $\rightarrow$ Similarity-sensitive surprisal

Shannon entropy $\rightarrow$ Similarity-sensitive entropy

A practical algorithm for MBR

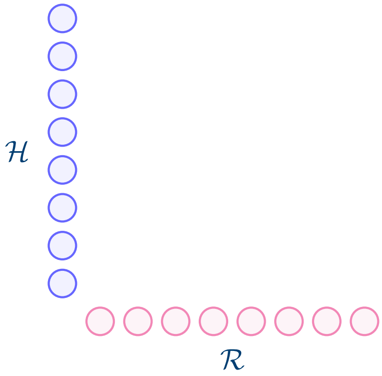
Generate candidates $\mathcal{H}$.

$\mathcal{H}$

Algorithm from Eikema and Aziz, 2020

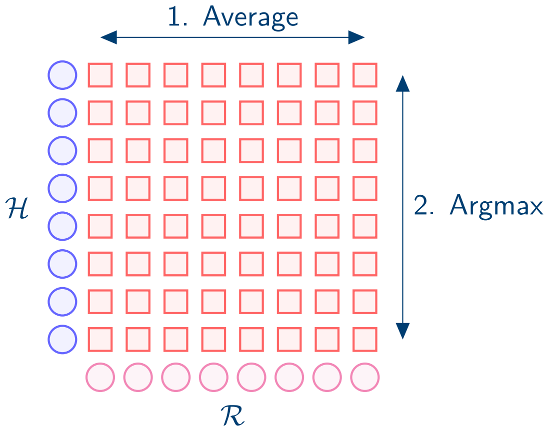
A practical algorithm for MBR

Generate pseudo-references $\mathcal{R}$.



A practical algorithm for MBR

Compute similarities. Average for each $y \in \mathcal{H}$. Take argmax.

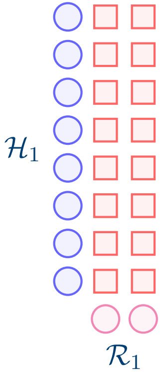


A practical algorithm for MBR

- **Problem**: this is slow - requires $\mathcal{O}(|\mathcal{H}||\mathcal{R}|)$ calls to $\mathcal{L}$.
- **Solution**: confidence-based pruning (Cheng and Vlachos, 2023)

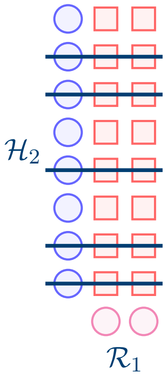
Confidence-based pruning for MBR

Start with hypothesis set $\mathcal{H}_1$ and initial pseudo-references $\mathcal{R}_1$.
Compute utilities for all pairs $y \in \mathcal{H}_1, \hat{y} \in \mathcal{R}_1$.



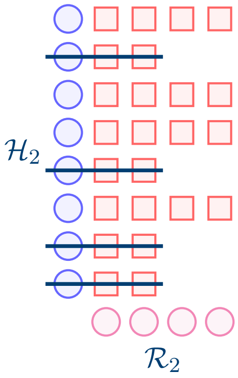
Confidence-based pruning for MBR

Apply a *pruning function* that returns $\mathcal{H}_2 \subseteq \mathcal{H}_1$.



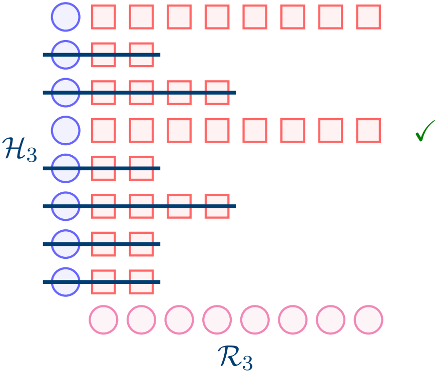
Confidence-based pruning for MBR

Construct $\mathcal{R}_2$ by appending new samples to $\mathcal{R}_1$. Compute utilities.



Confidence-based pruning for MBR

Repeat until one hypothesis left or maximum time step reached.
Return the hypothesis with the highest estimated utility.



Confidence-based pruning for MBR

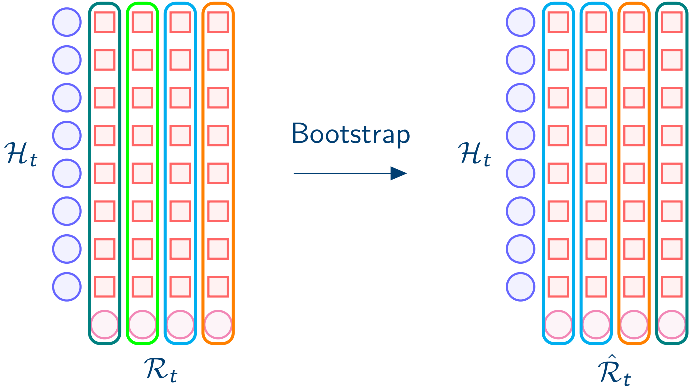
Pruning criterion: prune $y \in \mathcal{H}$ if $p(y)$ has less than $1 - \alpha$ chance of being the "true best".

$$
\begin{aligned}
& p(y \text{ is the true best}) \\
\approx \quad & p(y \text{ is the best in a bootstrap sample}) \\
\leq \quad & p(y \text{ is better than } y' \in \mathcal{H} \text{ in a bootstrap sample})
\end{aligned}
$$

where y' is set to be a candidate with highest utility in $\mathcal{R}_t$

Last step upper bound is because prob. of y winning is small when $\mathcal{H}$ is large. This removes the effect of set size.

Confidence-based pruning for MBR



Confidence-based pruning for MBR

- Experiments on de-en, en-et, tr-en.
- Returns to same result as full MBR 85% of the time with no quality drop.
- Single parameter - confidence threshold - controls quality/speed tradeoff
- Uses 12-15% as many calls to chrF++ and 3-5% for COMET.

Minimum Bayes risk decoding (MBR)

Why does MBR work?

- Returns sequences with probable **features**, not just high probability.
- MBR is **reference-based reranking** with **pseudo-references**. Want to score candidates $y \in \mathbf{y}$ with reference-based loss $\mathcal{L}(y, y^*)$ or $\mathcal{L}(y, y^*, x)$, but we don't have y^*.

Reranking methods: summary

Discriminative reranking	Requires no extra data
Noisy channel reranking	Can exploit monolingual data
Quality-based reranking	Needs human annotation for best results
Minimum Bayes risk	Needs human annotation for best results, slow

No cross-comparisons seem to exist...

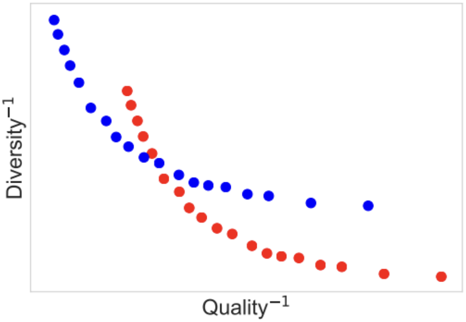
Candidate search

The candidate list need not be fixed...

- Monte Carlo tree search (Leblond et al., 2021)
- Genetic algorithm (Jon et al., 2023)
- Hypothesis recombination (Vernikos and Popescu-Belis, 2024)

Decoding for diversity

Methods which optimize quality and diversity. Evaluated on a *quality-diversity tradeoff curve*.



Language GANs Falling Short, Caccia et al., 2018.

Decoding for diversity

- Sampling methods
  - Probability-warping methods
  - Without-replacement sampling
- Sample-then-select methods

Sampling for diversity

Almost always, the main problem with ancestral sampling is low probability, low-quality generations.

Probability-warping methods besides temperature scaling, nucleus sampling, top-k? (Hewitt et al., 2022.)

- ϵ-sampling: set all tokens with less than ϵ prob. to 0 prob.
- η-sampling: combined with ϵ-sampling to also exclude tokens with $p(y_t) < \alpha \exp(\mathcal{H})$ prob., where $\mathcal{H}$ is the entropy of $p(y_t)$.

Without-replacement sequence sampling

Stochastic beam search (Kool et al., 2019).

- Use the **Gumbel top-k trick** to select the next beam continuations: add Gumbel noise z^i to the logprob of each next-token y^i.

$$x^i = \text{Uniform}(0, 1)$$
$$z^i = -\log(-\log(x_i))$$

- Run the standard beam search algorithm, except the perturbed logprobs are propagated in subsequent steps.
- Results in unbiased sequence sampling without replacement!

Sample-then-select methods

Get initial candidates $\mathbf{y}$. Select the subset $\mathbf{y}'$ which maximizes quality and diversity:

$$\arg\max_{\mathbf{y} \subset \mathbf{y}'} \Big(\sum_{y \in \mathbf{y}'} \mathcal{Q}(y) \Big) + d(\mathbf{y}')$$

where $\mathcal{Q}, d$ are quality and diversity functions, respectively.

This is a non-monotonic submodular function - NP-hard!

Sample-then-select methods

- Diverse beam search (Vijayakumar et al., 2016): Augments beam search with a dissimilarity objective.
- Determinantal beam search (Meister et al., 2021): Treat beam search next-token selection as a subdeterminant maximization problem which maximizes quality and diversity.
- Diverse MBR (Jinnai et al, 2024): Use MBR utility as the quality function.

Which generation method is right for you?

- For reranking or MBR candidate generation: prioritize quality if n is small. Prioritize diversity as n grows.
- For MBR pseudo-reference generation: objective requires a (possibly warped) unbiased estimate. ϵ-sampling with 0.02 is weirdly good (Freitag et al., 2023).
- Need diversity? Sample or use diverse decoding.

Conclusion

- When choosing a decoding method, consider:
  - What **data** you have
  - What **evaluation metrics** you have
  - Your compute budget
- LMs aren't perfect, but we can still get more out of them with good decoding!

Thanks!

jncc3@cam.ac.uk