RAG-Sequence: Top-k Approximation & Thorough Decoding

Wook-Shin Han

POSTECH

May 14, 2025

Contents

1. Overview

In the Retrieval-Augmented Generation (**RAG-Sequence**) model, the generation of a target sequence $y = (y_1, \ldots, y_N)$ conditioned on input x involves marginalizing over a latent variable z, which denotes the retrieved document.

Exact marginal likelihood:

$$p(y \mid x) \;=\; \sum_{z \in \mathcal{D}} p_\eta(z \mid x) \cdot p_\theta(y \mid x, z).$$

However, summing over all documents in $\mathcal{D}$ is intractable if $\mathcal{D}$ is huge (e.g., all of Wikipedia).

2. Top-k Approximation

Key Idea: Restrict the sum to the top-k most relevant documents according to $p_\eta(z \mid x)$. Then,

$$p(y \mid x) \approx \sum_{z \in \text{top-}k\left(p_\eta(\cdot|x)\right)} p_\eta(z \mid x) \cdot p_\theta(y \mid x, z).$$

Justification:

- Often, most probability mass of $p_\eta(z \mid x)$ lies in a small subset of documents.
- The retriever is fine-tuned to focus on these top-k.
- Empirically, it works well while remaining computationally feasible.

3. Gradient Derivation (Approx. Marginal)

The training loss is the negative log-likelihood of the *approximated* marginal probability:

$$\mathcal{L}(x, y) = -\log \sum_{z \in \mathcal{Z}} \Big[p_\eta(z \mid x) \cdot p_\theta(y \mid x, z) \Big], \quad \mathcal{Z} = \text{top-}k\big(p_\eta(\cdot \mid x)\big).$$

Let

$$A(z) := p_\eta(z \mid x) \cdot p_\theta(y \mid x, z).$$

Then

$$\mathcal{L}(x, y) = -\log \sum_{z \in \mathcal{Z}} A(z).$$

Gradient Derivation (continued)

Using

$$\nabla \log\Big(\sum_z A(z)\Big) = \frac{1}{\sum_z A(z)} \sum_z \nabla A(z),$$

we get:

$$\nabla \mathcal{L}(x, y) = -\frac{1}{p(y \mid x)} \sum_{z \in \mathcal{Z}} \nabla\Big(p_\eta(z \mid x) \cdot p_\theta(y \mid x, z)\Big).$$

By the product rule:

$$\nabla A(z) = \nabla\Big(p_\eta(z \mid x)\, p_\theta(y \mid x, z)\Big) = \nabla p_\eta(z \mid x) \cdot p_\theta(y \mid x, z) + p_\eta(z \mid x) \cdot \nabla p_\theta(y \mid x, z).$$

Thus,

$$\nabla\, \mathcal{L}(x, y) = -\frac{1}{p(y \mid x)} \sum_{z \in \mathcal{Z}} \Big[\nabla p_\eta(z \mid x) \cdot p_\theta(y \mid x, z) + p_\eta(z \mid x) \cdot \nabla p_\theta(y \mid x, z)\Big].$$

4. Remarks on Top-k

- Top-k approximation makes RAG feasible for large corpora, yet effective in practice.
- The better the retriever performance, the closer the sum over top-k is to the true marginal.
- We can train both retriever (p_η) and generator (p_θ) end-to-end.

Pipeline Overview

1. **Retrieval** (produces $p_\eta(z_k \mid \mathbf{x})$)
   Encode question $\mathbf{x}$, score documents, soft-max top–K.
2. **Beam search (per document)** (produces $p_\theta(y \mid \mathbf{x}, z_i)$)
   Expand beams token-by-token with decoder conditioned on single doc z_i.
3. **Marginalisation** (produces $p(y \mid \mathbf{x})$)
   Multiply each sequence score by its document prior and sum over docs.
4. **Selection**
   Pick answer $y^* = \arg\max_y p(y \mid \mathbf{x})$.

Stage 3: Marginalisation and Selection

$$p(y \mid \mathbf{x}) \;=\; \sum_{k=1}^{K} p_\eta(z_k \mid \mathbf{x})\; p_\theta\big(y \mid \mathbf{x}, z_k\big)$$

- Multiply the cached document prior by the sequence probability from each beam.
- Sum the weighted scores across all retrieved documents.
- Choose $y^* = \arg\max_y p(y \mid \mathbf{x})$.

All computations at a glance

Stage	Key operations	Quantity produced
Retrieval	Encode $\mathbf{x}$; ANN search; soft-max top-K scores	$p_\eta(z_k \mid \mathbf{x})$
Beam (per z_i)	Token-level softmax; beam pruning; accumulate p_θ	$p_\theta(y \mid \mathbf{x}, z_i)$
Marginalisation	Weighted sum of priors and sequence probs	$p(y \mid \mathbf{x})$
Selection	Arg-max over candidates	Final answer y^*

7. Worked Example (1/2): Retrieval & Beam Search

Question x: "What is the capital of France?" Top-$K = 3$ documents retrieved.

Stage 1 – Retrieval $\Longrightarrow p_\eta(z_k \mid \mathbf{x})$

Document	s_k	$p_\eta(z_k \mid \mathbf{x})$
z_1 : "Paris is the capital ..."	14.0	0.73
z_2 : "Lyon is a major city ..."	10.0	0.15
z_3 : "France, officially ..."	9.0	0.12

Stage 2 – Beam Search per doc $\Longrightarrow p_\theta(y \mid \mathbf{x}, z_i)$

From z_1	$\begin{cases} y_{1,1} = \text{"Paris"}, & p_\theta = 0.607 \\ y_{1,2} = \text{"The city Paris"}, & p_\theta = 0.301 \end{cases}$
From z_2	$y_{2,1} = \text{"Lyon"}$, $p_\theta = 0.670$ *(second beam pruned owing to low probability)*
From z_3	$y_{3,1} = \text{"Paris"}$, $p_\theta = 0.407$ *(only one high-scoring beam survives)*

Each p_θ is the (unnormalised) sequence probability obtained by multiplying token probabilities along the beam.

8. Worked Example (2/2): Marginalisation & Selection

Stagenbsp;3 – Marginalise across documents

$$p(y \mid \mathbf{x}) = \sum_{k=1}^{3} p_\eta(z_k \mid \mathbf{x})\; p_\theta\big(y \mid \mathbf{x}, z_k\big)$$

Candidate y	$p(y \mid \mathbf{x})$	Comment
"Paris" (from z_1, z_3)	$0.73 \times 0.607 + 0.12 \times 0.407 \approx 0.492$	high prior and good fit
"The city Paris"	$0.73 \times 0.301 \approx 0.220$	lower decoder score
"Lyon"	$0.15 \times 0.670 \approx 0.101$	low prior + wrong doc

Stagenbsp;4 – Select best answer

$$y^* = \arg\max_y p(y \mid \mathbf{x}) \;=\; \textbf{"Paris"}$$

Even though "Paris" appears in two documents, the marginalisation step weights each occurrence by its document prior, yielding the correct answer with the highest overall probability.

8. Observations

- **Complexity (rough):** Potentially $O(k^2 \times \text{beam\_size})$ forward passes.
- **Pros:** More accurate because it computes a *true* marginal (across top-k docs).
- **Fast Decoding:** A simpler method that avoids re-scoring sequences that never appear in each doc's beam.
- **Trade-off:** Thorough decoding can be slower but often yields better results; fast decoding is more scalable.

9. Why $O(k^2 \times \text{beam\_size})$?

Step 1: Beam Search per Document

- We have k documents.
- Each beam search produces beam_size candidate sequences.
- Total candidate sequences $= k \times$ beam_size.

Step 2: Re-evaluation (Marginalization)

- Each candidate $y_{i,j}$ must be re-scored under *all* k documents:

$$p(y_{i,j} \mid x) = \sum_{m=1}^{k} p_\eta(z_m \mid x)\, p_\theta\big(y_{i,j} \mid x, z_m\big).$$

- Hence, $\big(k \times \text{beam\_size}\big) \times k$ forward passes $= k^2 \times$ beam_size.

Overall Cost:

$$O\big(\underbrace{k \times (\text{beam search cost})}_{\text{Step 1}} + \underbrace{k^2 \times \text{beam\_size}}_{\text{Step 2}}\big).$$

In practice, the $k^2 \times$ beam_size re-scoring dominates.