Mahānveṣaṇa: A Knowledge-Enhanced RAG Framework for Semantic Inquiry into the Mahābhārata

Chirayush Mohanty (251110602)*　　**Roopesh Mangal (251040633)***

Shravani Agrawal (242110402)*　　**Srinidhi Balasubramanian (251110613)***

Varad Shinde (251110081)*

Abstract

What are we doing?

The Mahābhārata, with its complex narrative structure, poetic style, and character reference complexity, presents a challenge for question-answering and verse retrieval. Standard Retrieval-Augmentation Generation models (RAG) often fail/struggle with these challenges. Our work (Mahānveṣaṇa) is a knowledge enhanced RAG framework to address this issue. We present a comparative analysis of SoTA models for verse retrieval.

We integrated the Mahānāma entity linking data to resolve the complexity of entity ambiguity, and by using chapter boundaries we create a chapter inter-relatedness graph. Using the notion of contrastive learning, we tried to finetune the best performing SBERT model to improve the retrieval which yielded marginal improvements.

We created a custom validation and evaluation dataset of over 2000 query-verse pairs to evaluate the performance of our model. Results showed that integration of external knowledge yielded significant increase $\tilde{1}0\%$ in the recall scores for open source embedding models in validation phase and $\tilde{5}\%$ in testing phase.

1 Introduction

The Mahābhārata, one of the two epics traditionally known as *Itihāsa* is the world's longest epic poem and holds great relevance and importance within the Indian Knowledge Systems. With over 80,0000 verses, Mahābhārata comprises of a complex web of interwoven narratives philosophical discourses, and poetic references which makes it challenging for standard NLP models to perform well in question-answering or verse-retrievals. Some of the complexities presented by the Mahābhārata:

- **Variability and Ambiguity in Entity names [1]**: Throughout the epic, most entities are referred to by multiple names (variability) and the same name could also refer to different entities (ambiguity) depending on the context. For example, within a few verses, Yudhiṣṭira is referred to as Kaunteya and Ajataśatru whereas the term Kaunteya can refer to any of the three sons of Kunti. Conventional RAG retrieval architectures that rely on semantic embeddings do not capture this complexity.
- **Non-continuous and nested narratives**: Aside from the central plot and story line which is in the form of a conversation between Vaiśaṃpāyana and Janamejaya, Mahābhārata presents hundereds of *Ākhyanas* and *Upākhyānas* (sub-stories) embedded within as conversations and narrations by various characters within the main story line. This nesting can have many layers

*All authors contributed equally to this work.

and hence causing breaks in the narrative. From an implementation point of view, the context needed to answer a query is often not located in a single block of text and the entire context is important for a detailed and accurate answer to the query.

- **Sanskrit non-translatables and poetic devices** Some terms as *Mokṣa*, *Karma*, *Dharma* etc. are considered Sanskrit non-translatables and often queries with such terms do not match accurately with their corresponding english translations.

1.1 Performance and drawbacks of LLMs and Naïve RAG

Due to the above mentioned reasons, LLMs often fail to answer questions or retrieve verses from the Mahābhārata. Preliminary random experiments with ChatGPT (GPT-5 and GPT-4o) reveal the following shortcomings as a result of its inherent design-

- **Hallucination**: This is a major concern and a fundamental challenge for all large language models and recent studies claim that this is an inherent result of the evaluation methods used to train LLMs. As shown in Figure 1, on asking a question to retrieve verses relevant to a particular story, it shows certain verses which seem plausibly true but actually do not exist anywhere in the text!
- **Fixed top-k hyperparameter**: Naïve RAG systems employ the hyperparameter k to retrieve the top k similar chunks to the query (or alternatively, chunks with similarity above a threshold value). However, for practical purposes, the expected retrieved chunks can be lesser or greater than k (alternately, all relevant chunks may not have high similarity values). For example, in Figure 1, three verses are returned whereas the story in the actual texts extends to more than 30 verses.

2 Problem Statement

The objective of this work is to design a RAG based QnA and Verse Retrieval system that can retrieve all original verses that are relevant to a given query in English and subsequently generate a coherent answer based on the above.

For instance, given a query "What was the conversation between Janaka and Sulabha", the system shall return the list of **all** verses, in this case, verses 12.308.1 to 12.308.191 and an answer summarizing or explaining the above.

We formulate the task as follows:

Given a natural language query Q (in English), the task is to retrieve all and only relevant verses (Sanskrit) leveraging auxiliary knowledge sources (translations, entity relationships, speaker information, narrative graphs etc.)

v_i from a large corpus of Sanskrit verses $\mathcal{C} = \{v_1, v_2, \ldots, v_M\}$ such that each retrieved verse (or verse block) is semantically and contextually relevant to the query.

Given,

$$Q, \quad \mathcal{C} = \{v_1, v_2, \ldots, v_M\}, \quad \text{and} \quad \mathcal{K}$$

Retrieve,

$$\mathcal{V}_R = \{v_1, v_2, \ldots, v_n\} \subseteq \mathcal{C}$$

such that

$$\forall v_i \in \mathcal{V}_R,\ v_i \text{ is relevant to } Q, \quad \text{and} \quad \forall v_j \in \mathcal{C} \setminus \mathcal{V}_R,\ v_j \text{ is not relevant to } Q.$$

In other words, the retrieved set $\mathcal{V}_R$ achieves both **completeness** (all relevant verses are extracted) and **purity** (no irrelevant verses are included), with relevance determined jointly from the verse content and the auxiliary knowledge $\mathcal{K}$. Here, N is variable and adaptive, depending on the scope and narrative breadth of the query Q.

3 Methodology

The complete retrieval pipeline thus consists of:

Parva	Chapters	Words
1	234	215654
2	81	72262
3	315	307730
4	72	55814
5	193	186620
6	123	145758
7	203	257036
8	96	141814
9	65	84954
10	18	19727
11	27	20488
12	365	422797
13	168	260352
14	92	78827
15	39	29547
16	8	7899
17	3	3005
18	6	8545
Total	2108	2318829

Table 1: Parva-Chapter Division in Corpus

1. Dense retrieval using sentence embeddings.
2. Entity extraction and TF–IDF relevance.
3. Graph-based propagation using chapter similarity.
4. Fusion of semantic and entity scores.
5. Clustering and filtering to refine final context for generation.

4 Dataset

For evaluating our model, we created and used a pseudo-synthetic dataset consisting of query-verse for validating and testing. We used the Gita Press index in Hindi which consisted of descriptions of each chapter of the Mahābhārata to source chapter descriptions. Next, we mapped each description to the corresponding chapter in the Itihasa dataset which is a Sanskrit-English translation corpus containing over 93000 Sanskrit shlokas and their translation. This helped align the summaries with the original source verses. To be able to utilise natural-language for queries, the Hindi chapter descriptions were translated into English using Google Translate API. These translated chapter descriptions were used to construct natural language queries using an LLM. This resulted in a dataset with over 2000 query-verse pairs which we split into a 25:75 validation-test set to conduct an evaluation of the retrieval model.

5 Retrieval Architecture

5.1 Embedding Models

We experimented with open source SBERT models for retrieval of chapters for query. Initial exploration resulted in MPNET giving the best results.

We take this SBERT too finetune it on the Mahabharata data using the notion of contrastive learning.

Further, in order to benchmark our improvements, we also test out OpenAI's text-embedding-ada as the embedding model.

5.2 Finetuning SBERT

We finetune SBERT using a contrastive learning objective tailored to the narrative structure of the Mahabharata. The cleaned corpus is segmented into overlapping windows, from which we construct around

30k supervised contrastive pairs. Each anchor is paired with one positive example drawn from the same chapter or adjacent narrative region, while multiple negatives—typically 5–15 per anchor—are sampled from semantically distant chapters. This yields a rich contrastive dataset that captures both intra-chapter coherence and inter-chapter divergence.

Training follows the standard SBERT contrastive paradigm, where all samples in a batch implicitly serve as negatives for one another. We employ a contrastive ranking loss that minimizes the distance between anchor–positive pairs while maximizing separation from negatives. Finetuning is performed for 3–5 epochs, with a moderate learning rate, a batch size suited for contrastive objectives, and sequence lengths adjusted for long narrative segments. These choices follow established best practices for domain adaptation of transformer encoders.

Across training, the loss exhibits a stable, monotonic decline, with the most significant improvements occurring in the first two epochs and convergence observed by the fourth. This behavior indicates that the model rapidly learns the high-level semantic structure of the text and subsequently refines fine-grained distinctions between chapters. The resulting encoder shows markedly improved clustering of related passages and forms the backbone of our retrieval pipeline.

5.3 Entities and Graph-based Score Propagation

A major limitation of standard retrieval models (e.g., TF-IDF or sentence-transformer embeddings) is that they treat chapters as independent text units. However, in the *Mahabharata*, meaning is distributed across chapters — the same entities (characters, locations, events) recur across contexts, forming implicit narrative links. To capture these relationships, we design an **entity-based similarity graph** connecting chapters through shared entities and thematic proximity.

Graph Construction

Let there be N chapters and M unique entities. Each chapter i is represented by an *entity frequency vector*

$$\mathbf{x}_i = [f_{i1}, f_{i2}, \dots, f_{iM}],$$

where f_{ij} denotes the normalized frequency of entity j in chapter i.

We compute **TF-IDF weighting** for each entity as

$$TFIDF_{ij} = TF_{ij} \times \log\left(\frac{N}{n_j}\right),$$

where n_j is the number of chapters containing entity j.

The **chapter similarity graph** is then defined as a weighted, undirected graph

$$G = (V, E, W),$$

where V are chapters, and

$$W_{ij} = \cos(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\|\mathbf{x}_i\| \, \|\mathbf{x}_j\|}$$

is the *edge weight* denoting entity-level similarity between chapters i and j.

Direct Relevance

Given a user query decomposed into a set of entities $Q = \{e_1, e_2, \dots, e_k\}$ with associated weights (confidence score of the entity e_i being present in the query) $w = [w_1, w_2, \dots, w_k]$, we first compute the **direct relevance** for each chapter:

$$r_i = \sum_{e_j \in Q} w_j \, X_{i,e_j},$$

where X_{i,e_j} is the TF-IDF score of entity e_j in chapter i.

Consecutive entity IDs are grouped to enforce structural coherence. Let

$$G = \{g_1, g_2, \dots, g_m\}$$

be the set of groups.

Define the coverage score for chapter i:

$$\gamma_i = \frac{|\{g_k \in G : \text{chapter } i \text{ contains an entity from } g_k\}|}{|G|}.$$

The group-adjusted direct relevance becomes:

$$r'_i = r_i \cdot \gamma_i.$$

Graph-Based Relevance Propagation

$$\mathrm{Score}(j) = \sum_{i \in \mathrm{TopK}} \mathrm{Rel}(i)\, \mathrm{sim}(j, i)\, \mathbf{1}[\text{chapter } j \text{ contains a query entity}]\,.$$

Score Normalization

To stabilize scores:

$$\hat{p}_i = \frac{1}{1 + \exp(-\alpha(p_i - \mu))},$$

where μ is the mean of p_i.

Score Fusion

Embedding similarity and graph relevance are fused via:

$$Final_i = a \cdot s_i + b \cdot \hat{p}_i,$$

weights were set to $a = 0.9$, $b = 0.1$ after observation from validation data.

5.4 Clustering-Based Post-Filtering

We take the top-N chapters under fused score and run KMeans with $k = 2$ clusters. Let c_i be the cluster assignment.

We select the cluster with highest average score:

$$c^{=\arg\max_c \mathbb{E}[s_i | c_i = c].}$$

The final retrieved set is:

$$BestDocs = \{i : c_i = c^{\}.}$$

This removes noisy false positives and tightens the retrieved set.

6 Experiments

6.1 Finetuning

6.2 Retrieval

Model	Entity Scoring	Top-k	Precision	Recall	Hits	nDCG	MRR
Model A	Yes/No	5	0.00	0.00	0.00	0.00	0.00
Model B	Yes/No	5	0.00	0.00	0.00	0.00	0.00
Model C	Yes/No	5	0.00	0.00	0.00	0.00	0.00
Model D	Yes/No	5	0.00	0.00	0.00	0.00	0.00
Model E	Yes/No	5	0.00	0.00	0.00	0.00	0.00
Model F	Yes/No	5	0.00	0.00	0.00	0.00	0.00

Table 2: Comparison of retrieval models

6.3 Generation

Due to the lack of ground truth of long form answers, we do not evaluate the final generated answers quantitatively. However a qualitative evaluation of a sample successful and unsuccessful cases are presented in Appendix B.

7 Limitations

While Mahānveṣaṇā presents a novel question-answering method for the Mahābhārata, the improvement in performance after fine-tuning was only marginal. Further, it relies heavily on knowledge graphs such as the entity-linking dataset. Thus, incomplete entity information would largely limit the performance of the model. The nature of the data also means that for finetuning SBERT, the similarity of verses makes it hard to create good negative examples. The chapter-level graph may miss connections between individual verses, and chapters with incomplete entity information can make retrieval less accurate. Also, the way relevance is spread across chapters is based on rules instead of being learned from data, so it may not work well for or complicated queries.

8 Further Scope

While Mahānveṣaṇā showed a marginal improvement on verse retrieval and question-answering for the Mahābhārata, several directions remain open for further enhancement. One is developing other knowledge sources, as well as more narrative-graph information like causal chains and linking across episodes, etc. could allow for more complex query addressing. It also highlights the need for other translations and databases/knowledge graphs to facilitate this work.

Beyond the present implementation, a promising direction for future work is the extension of our chapter-level entity graph into a fully learnable graph neural network (GNN)–based retrieval module. Also, by integrating verse-level nodes into the graph, hierarchical retrieval can be facilitated, where chapter-level relevance can also guide more precise verse extraction.

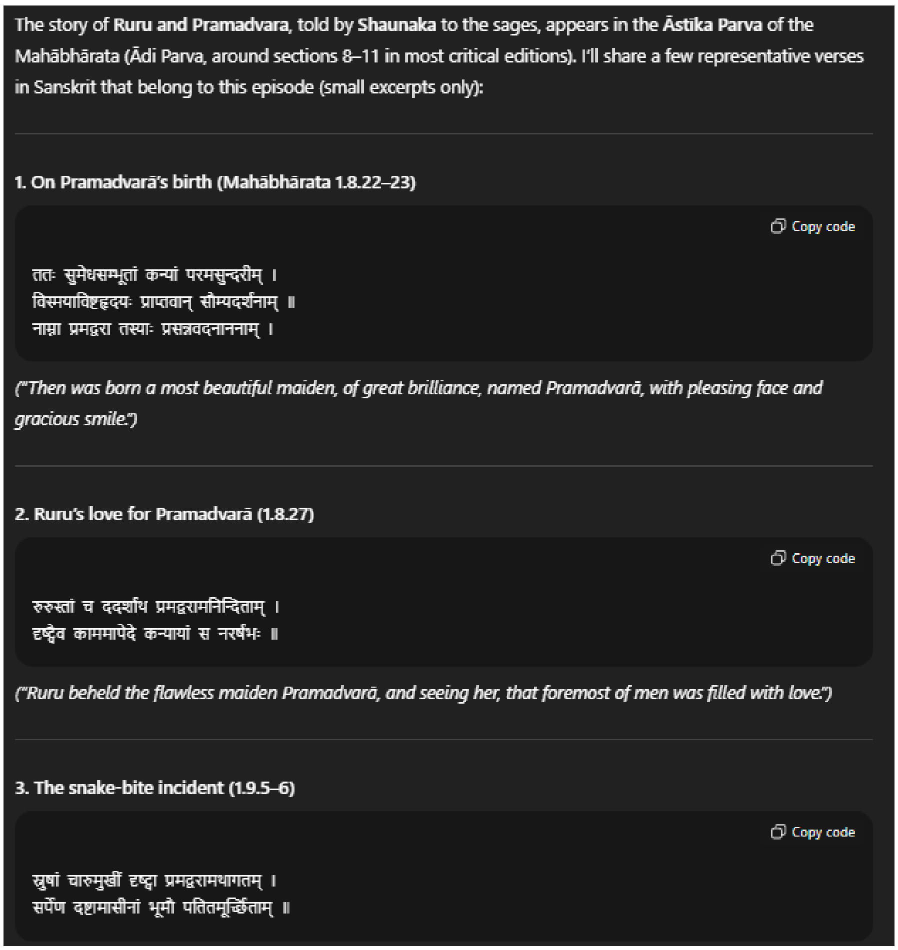
With more work, this can also be extended to other Itihāsa-Purāṇa texts or to multilingual user queries.

9 Conclusion

We developed Mahānveṣaṇā, a knowledge-enhanced RAG framework designed to address challenges in verse retrieval in the Mahābhārata. We used the Mahānāma entity-linking dataset, used chapter information, and used an entity-linking graph, to enable the retrieval system to retrieve verses relevant to a query in English. We also fine-tuned SBERT models on Sanskrit-English parallel data and created an evaluation dataset of over 2000 query-verse pairs. We found that augmenting retrieval with external knowledge yielded a 10% improvement in recall during validation and 5% on the held-out test data, across embedding models. Overall, Mahānveṣaṇā shows that structured knowledge and narrative-aware propagation can be used to improve performance on complex texts, laying the groundwork for more advanced retrieval systems for Itihāsa and other Sanskrit corpora.

References

[1] Sujoy Sarkar, Gourav Sarkar, Manoj Balaji Jagadeeshan, Jivnesh Sandhan, Amrith Krishna, and Pawan Goyal. Mah\={a} n\={a} ma: A unique testbed for literary entity discovery and linking. *arXiv preprint arXiv:2509.19844*, 2025.



The story of **Ruru and Pramadvara**, told by **Shaunaka** to the sages, appears in the **Āstīka Parva** of the Mahābhārata (Ādi Parva, around sections 8–11 in most critical editions). I'll share a few representative verses in Sanskrit that belong to this episode (small excerpts only):

1. On Pramadvarā's birth (Mahābhārata 1.8.22–23)

```
ततः सुमेधसम्भूतां कन्यां परमसुन्दरीम् ।
विस्मयाविष्टहृदयः प्राप्तवान् सौम्यदर्शनाम् ॥
नाम्ना प्रमद्वरा तस्याः प्रसन्नवदनाननाम् ।
```

("Then was born a most beautiful maiden, of great brilliance, named Pramadvarā, with pleasing face and gracious smile.")

2. Ruru's love for Pramadvarā (1.8.27)

```
रुरुस्तां च ददर्शाथ प्रमद्वरामनिन्दिताम् ।
दृष्ट्वैव काममापेदे कन्यायां स नरर्षभः ॥
```

("Ruru beheld the flawless maiden Pramadvarā, and seeing her, that foremost of men was filled with love.")

3. The snake-bite incident (1.9.5–6)

```
स्नुषां चारुमुखीं दृष्ट्वा प्रमद्वरामथागतम् ।
सर्पेण दष्टामासीनां भूमौ पतितमूर्च्छिताम् ॥
```

Figure 1: ChatGPT when asked "Can you retrive the exact sanskrit verses from mahabharata for the story of Shaunaka and Ruru"