A Novel LLM-Based Approach for Automated Seerah-Hadith Mapping: Connecting Islamic Historical Narratives Through Vector Search and Semantic Analysis

Mushfiqur Rahman Talha
Greentech Apps Foundation
mushfiq.talha@gtaf.org

Mohammad Galib Shams
Greentech Apps Foundation
galib.shams@gtaf.org

Nabil Mosharraf Hossain
Greentech Apps Foundation
nabil@gtaf.org

Riasat Islam
Greentech Apps Foundation
riasat.islam@gtaf.org

Abstract

Seerah and Hadith are essential sources of Islamic knowledge, but there has been limited research on systematically linking these two areas. This paper introduces the Seerah-Hadith Mapping project, which uses Large Language Models (LLMs) to map related passages between Seerah and Hadith. By adding new connections between these texts, this approach builds on understanding and helps make Islamic knowledge more accessible to those without specialized knowledge in Islamic studies.

1 Introduction

In Islamic studies, Seerah[1] and Hadith[2] are vital sources for understanding the life and teachings of Prophet Muhammad (Peace Be Upon Him, PBUH). Seerah, or the biography of the Prophet (PBUH), provides detailed insights into his character, actions, and the historical context at the time. However, Hadith focuses on the sayings, actions, and approvals of the Prophet (PBUH), offering direct guidance for Islamic law and daily life. Together, Seerah and Hadith are essential for interpreting Islamic teachings, as they help clarify the context of Quranic revelations and demonstrate their application in real-life situations.

Linking Seerah and Hadith texts through automated methods presents several challenges. Seerah is organized as a chronological narrative, while Hadiths are typically focused on specific issues and are presented in a more fragmented way, often without clear time markers. Additionally, many Hadiths lack detailed context, making it difficult for non-experts to connect them with specific events in the Prophet (PBUH)'s life. Some works have been done on Islamic literature, such as embedding search in Quranic texts[3], building domain-specific LLMs in Islamic worldview[4], semantic similarity on Holy Quran translations using S-BERT[5] etc.

The Seerah-Hadith Mapping project tackles these challenges by using LLMs to automatically link related Hadiths to specific events in the Seerah. This method supports scholarly research by creating structured connections between Seerah narratives and Hadiths, making these important texts easier to understand for non-experts in Islamic history and law.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

2 Method

Throughout our whole working procedure we have used the databases of books *The Sealed Nectar: Ar-Raheeq Al-Makhtum*[1], and *The Prophetic Timeline: The Life of Prophet Muhammad (PBUH)*[6] for Seerah. For Hadith, we used the database obtained from sunnah.com[7].

2.1 Pre-processing Seerah and Hadith documents

The first step in this task was to preprocess the Seerah and Hadith text database to prepare the texts for analysis using LLM reasoning and vector search. The steps are below:

1. **Content cleaning:** The Seerah and Hadith text database contained extra elements that could disrupt effective natural language processing and vector search. We removed these common phrases, repeated annotations, and references such as "See previous hadith", "Similarly as no", or specific names like "Sahih", "Bukhari", and "At-Tirmidhi" from both Seerah and Hadith texts for better search results. Also, special characters, symbols, and annotations were removed. Then, the texts were converted to lowercase, and extra spaces or unnecessary punctuation were removed.
2. **Seerah paragraph segmentation:** After cleaning, the Seerah texts were split into meaningful paragraphs by double newline characters (`\n\n`). Each paragraph was tokenized using `word_tokenization` function provided by *NLTK*[8] library. Any punctuation or symbols that remained after content cleaning were further removed during this step to isolate relevant words. We tried to maintain at least 200 characters per paragraph, to maintain the context of the events.
3. **Tokenization and vector embedding creation:** In the next step, the tokenization of the Hadith texts was carried out using the `word_tokenize` function provided by the *NLTK*[8] toolkit. This process involved breaking down each Hadith into individual tokens, with a view to convert the natural language text into a format suitable for embedding models. After this, they were passed through the *bge-large-en-v1.5*[9] embedding model, to get high-dimensional vector representations (embeddings).
4. **Vector storage:** To facilitate fast and efficient search across the vast space of Hadith embeddings, the vectors were stored using the Hierarchical Navigable Small World (HNSW)[10] graph data structure. HNSW is well-suited for approximate nearest neighbor (ANN) search.

2.2 Retrieving similar Hadiths for Seerah paragraphs

For each of the Seerah paragraphs found through process (*Seerah paragraph segmentation*)2, we carried out the process (*Tokenization and vector embeddings creation*)3 to find out the vector embeddings for each Seerah paragraph. Then, for each paragraph, we conducted an Approximate Nearest Neighbor (ANN) search on the HNSW index created with the Hadith embeddings and took 10 related Hadiths. In the Hadith database, some of the Hadiths are just reference texts to other Hadiths. For optimization, those reference Hadiths were marked and we didn't conduct the next phases for those Hadiths.

2.3 Figuring out the type of matching

We designed a chain of 3 subsequent prompts using the **Chain of Responsibility**[11] design pattern. Each of the elements of the chain had 2 functions. One function was for creating the prompt using the Seerah paragraph and Hadith. Another function was needed to figure out the match type from current prompt response, to go to the next prompt. The prompts were executed in a chain, with each step refining the result based on the previous one. Below is a detailed description of the methodology.

1. **Initial match determination:** The first prompt checks the primary match type between a Seerah paragraph and a Hadith. The result is provided in JSON format with two keys:
   - `"match"` having values `"complete"`, `"partial"`, or `"no"`.
   - `"explanation"` containing a brief description of the matching result.

If the result is `complete`, we return the match categories as `complete_from_query_1`. If the result is `no`, the process ends with the match category `no`. Otherwise, 2nd prompt is executed on this pair of Seerah and Hadith to find their match type.

2. **Exact event match:** The second prompt examines whether an exact match exists between a Seerah event and a Hadith event. It gives the following JSON output:
   - `"match"` with values `"yes"`, `"no"`, or `"maybe"`.
   - `"explanation"` providing reasoning of the matching type.
   - `"event_title"` providing the title of the matched event.

   If the match is `no`, the process terminates with the match category `no`. Otherwise, 3rd query is executed for this pair of Seerah paragraph and Hadith.
3. **People and time matching:** The final prompt checks if the same people were involved in the event and if it occurred at the same time. This check is carried out for filtering out wrong `yes`'s from the previous prompt output. The output includes:
   - `"same_people_and_time"`: with values `"yes"` or `"no"`.
   - `"explanation"`: a description of the result.

   If the output from 2nd query was `yes`:
   - Output `yes` from 3rd query is considered as `complete` match.
   - Output `no` from 3rd query is considered as `high_priority_partial` match.

   If the output from 2nd query was `maybe`:
   - Output `yes` from 3rd query is considered as `high_priority_partial` match.
   - Output `no` from 3rd query is considered as `low_priority_partial` match.

Depending on the above mentioned propmpts, we categorized the types of matches in 5 categories which are: `complete_from_query_1`, `complete`, `high_priority_partial`, `low_priority_partial`, `no`. We considered `complete_from_query_1` and `complete` matches as proper completes and `high_priority_partial` matches are taken into consideration for further manual review.

For the output from LLM reasoning, we used the LLaMa 3 model (70b parameters)[12] API, provided by GroqCloud[13]. We used the temperature settings equal to 0 for retrieving deterministic outputs from the LLM.

3 Testcase preparation

We took a sample of 100 Seerah paragraphs, on which we created the sample testcase. For each of the paragraphs, we found 10 related Hadiths, creating a total of 1000 Seerah-Hadith pairs. Then we carried out the chain of prompts mentioned in Section 2.3, and retrieved the pairs with `no` match. Among the 1000 outputs, we manually inspected 100 outputs that have an inference output of `no`, and found that all were correct inferences. So, all pairs that had the match type `no` were kept in the test set. We also kept the `partial` matches as is. For the `complete` matches, we manually inspected all those Seerah-Hadith pairs and verified the match types. This way, a test set of a total of 1000 Seerah-Hadith pairs was created.

Later 50 more random Seerah paragraphs were added to the test set, to verify that our procedure is not biased to the previous test set of 100 sample Seerah paragraphs. Our final test set had a total of 150 Seerah paragraphs, i.e. a total of 1500 Seerah-Hadith pairs.

4 Accuracy calculation

In our approach, from the inference of the LLM, there were a huge amount (approximately 600+) of `no` match types. Manually verifying all these `no` matches was difficult. Similarly, for `partial` matches, there were around 300 pairs. So we skipped verification of the `partial` and `no` match types. On the other hand, the number of `complete` match types were around 120. So, it was possible for us to verify all these `complete` matches manually.

Since, we could only verify the `complete` match types, we only calculated the accuracy for complete matches. So, our accuracy measurement of completes match types is as follows:

$$correct\_accuracy = \frac{correct\_completes}{total\_completes}$$

Where, $correct\_completes$ = Total completes (found through LLM inference) which are correct according to the test set. And $total\_completes$ = Total pairs that had complete matching type according to the LLM inference.

5 Result analysis

5.1 Result analysis on test set

We conducted 2 experiments on the test set. The results of this test set are as follows:

Metrics	Experiment 1	Experiment 2
Total completes found	119	121
Incorrect completes	5	6
Complete accuracy(4)	95.8%	95.04%
Paragraphs having at least 1 complete match	35	36
Paragraphs (among above found), should have no complete match	3	3
Paragraphs accuracy	91.43%	91.67%

Table 1: Experiment results

5.2 Result analysis on final output

After testing the sample test set, we performed the approach mentioned in Section 2. There were a total of 1660 Seerah paragraphs, i.e. a total of 16600 Seerah-Hadith pairs. After completing the inference of the LLM, we found a total of 1814 Seerah-Hadith pairs as the `complete` match type. A total of 1093 pairs were marked as `high priority partials`. A total of 1732 pairs were marked as `low priority partials`. A total of 22 Hadiths were marked as `references` to other Hadiths. And a total of 11332 pairs were marked as `no` match type. While receiving outputs from LLM, for the remaining 7 pairs of Seerah and Hadith pairs, we got `JSONDecodeError`, since we wanted the output from LLM in JSON format, and in these 7 cases, the LLM failed to generate the JSON formatted output. So, we tracked these 7 pairs separately to manually inspect the match type.

6 Limitations of our approach

In our approach, the model used to create embeddings [9], at time of experimentation ranked 42nd on the Massive Text Embedding Benchmark (MTEB)[14] leaderboard. This suggests there may be more suitable embedding models that could better capture the semantic meaning. For vector search, we used the HNSW[10] data structure, which allows for faster searching of similar Hadiths by balancing accuracy with speed. However, HNSW provides approximate rather than fully accurate results to improve inference speed. Enhancements could be made to improve the accuracy in verifying match types, as discussed in Section 2.3. Additionally, improvements could be applied to LLM prompting; for this, the prompt chain detailed in Section 2.3 could be refined to achieve better prompt accuracy. Lastly, we were unable to calculate the accuracy of other matching types. Including these calculations would have provided a more complete assessment of our method's effectiveness.

7 Conclusion

This paper introduced the Seerah-Hadith Mapping approach, which uses Large Language Models (LLMs) and vector search techniques to create connections within a Seerah and Hadith text database. By preprocessing the texts, the system prepared the data for embedding and similarity-based retrieval

using the HNSW data structure. This approach has proven efficient in retrieving relevant Hadiths for Seerah paragraphs, achieving high accuracy in identifying complete matches.

Nonetheless, some limitations remain, such as the trade-off between speed and accuracy in the approximate nearest neighbor (ANN) search, and the need for manual verification for partial or no-match types. Future enhancements could involve using more advanced embedding models and improved prompting techniques for LLMs, which would likely improve the system's performance. Despite these challenges, this approach lays a strong foundation for integrating core Islamic texts, aiding in a deeper understanding of Seerah and Hadith.

References

[1] S.-R. Al-Mubarakpuri, *The Sealed Nectar: Ar-Raheeq Al-Makhtum*. Darussalam Publishers, 2014.

[2] M. Madnī and I. Madani, *The Preservation of Hadith: A Brief Introduction to the Science of Hadith*. Madania Publications, 2010. [Online]. Available: https://books.google.com.bd/books?id=ZEjbngEACAAJ

[3] M. A. Alqarni, "Embedding search for quranic texts based on large language models." *Int. Arab J. Inf. Technol.*, vol. 21, no. 2, pp. 243–256, 2024.

[4] S. Patel, H. Kane, and R. Patel, "Building domain-specific llms faithful to the islamic worldview: Mirage or technical possibility?" *arXiv preprint arXiv:2312.06652*, 2023.

[5] T. Afzal, S. A. Rauf, and Q. Majid, "Semantic similarity of the holy quran translations with sentence-bert," in *2023 20th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*. IEEE, 2023, pp. 285–290.

[6] M. R. . D. F. Asim Khan, Mike Cooper (Narrator), *The Life of Prophet Muhammad*. Muslim Research & Development Foundation, 2011. [Online]. Available: https://www.goodreads.com/book/show/18630199-the-life-of-prophet-muhammad

[7] [Online]. Available: https://sunnah.com

[8] S. Bird and E. Loper, "NLTK: The natural language toolkit," in *Proceedings of the ACL Interactive Poster and Demonstration Sessions*. Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 214–217. [Online]. Available: https://aclanthology.org/P04-3031

[9] S. Xiao, Z. Liu, P. Zhang, and N. Muennighoff, "C-pack: Packaged resources to advance general chinese embedding," 2023.

[10] Y. A. Malkov and D. A. Yashunin, "Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 4, pp. 824–836, 2020.

[11] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994.

[12] "Llama 3 70b: Open source language model." [Online]. Available: https://github.com/meta-llama/llama3

[13] I. Groq, "Groqcloud: Ai acceleration software," 2024, accessed: 2024-11-03. [Online]. Available: https://groq.com/products/groqcloud

[14] N. Muennighoff, N. Tazi, L. Magne, and N. Reimers, "Mteb: Massive text embedding benchmark," *arXiv preprint arXiv:2210.07316*, 2022. [Online]. Available: https://arxiv.org/abs/2210.07316