Group10: Timeline Analysis for Ramayana.

Ashutosh Patel (21111018) Deepak B Hegde (21211402)
Harishchandra Patidar (21111029) Mandar Dhake (21111405)
{ashutoshp21, deepakbh21, hpatidar21, mkdhake21}@iitk.ac.in
Supervisor: Prof. Arnab Bhattacharya
Indian Institute of Technology Kanpur (IIT Kanpur)

April 27, 2022

Abstract

Temporal information retrieval is a rapidly growing branch of natural language processing and information extraction, due to numerous applications such as question answering and summarization systems. The detection of events, states, temporal expressions and their relations provides a rich source of temporal information, and acts as the representation of real world information in text. This has two-fold implications, first, that the representation mechanism depends on the syntactic and semantic properties of the language, and second, that in order to create systems that use this information, large amounts of annotated data are a prerequisite.[2]

We show how one can perform temporal ordering of events even when popular time events are unavailable such as in ancient historical literature such Ramayana and Mahabharata.

1 Introduction

Time clearly plays a central role in any information space, and it has been studied in several areas like information extraction, topic-detection, question-answering, query log analysis, and summarization. Time and temporal measurements can help recreating a particular historical period or describing the chronological context of a document or a collection of documents.[4]

The Ramayana is one of the largest ancient epics in world literature. It consists of nearly 24,000 verses (mostly set in the Shloka), divided into seven kaṇḍas, the first and the seventh being later additions. It belongs to the genre of Itihasa, narratives of past events, interspersed with teachings on the goals of human life. Scholars' estimates for the earliest stage of the text range from the 7^{th} to 4^{th} centuries BCE,with later stages extending up to the 3rd century CE. [9]

Our problem statement is to determine the syntactic and semantic properties, such as tense, parts of speech, etc., of the Ramayana data-set in Hindi and use the information retrieved to perform timeline analysis and temporal ordering of the events.

2 Related Work

Temporal expressions mentioned in text documents can be grouped into four types according to Time-ML [5] , the standard markup language for temporal information: date, time, duration, and set. While duration expressions are used to provide information about the length of an interval (e.g., "three years" in they have been traveling through the U.S. for three years), set expressions inform about the periodical aspect of an event (e.g., "twice a week" in she goes to the gym twice a week). In contrast, time and date expressions (e.g., "3 p.m." or "January 25, 2010") both refer to a specific point in time – though in a different granularity. An interesting key feature of temporal information is that it can be normalized to some standard format. Assuming a Gregorian calendar as representation of time, expressions of time and date can be directly placed on a timeline. A date is then typically represented as YYYY-MM-DD, e.g., the expression "January 25, 2010" is normalized to "2010-01- 25".

Though there are multiple time representations, most temporal ordering is based on ordering the exact time or date; however, in ancient historical literature such as the Ramayana and Mahabharata, we rarely find a specific timeline mentioned in the literature.

There is a need to solve temporal ordering for ancient historical literature because it can serve as a foundation for a variety of other applications such as question answering and summarization systems.

There has been very little progress in this particular area of research for Ramayana. Our objective is to perform timeline analysis of the Ramayana and provide temporal ordering of events based on a query or the entire data-set.

3 Proposed Idea

We propose to solve the problem in two stages, with the first task being to retrieve the parts of speech and tenses of all the kaand and sarg. The second stage would be to use the retrieved semantic and syntactic properties to temporally order the events based on the properties of a particular event. We will return a list of events that do not occur in the existing order for the entire data-set.

We also propose ranking the events based on whether they occurred in the past, present, or future ***for any given query***. We will analyse the output with the knowledge graph to gain a better understanding of the results obtained from the aforementioned approach.

4 Methodology

The research work is relatively new in this field and there are hardly any preprocessed data-sets available for Ramayana. Though we found the Sanskrit to English translation of the Ramayana from the itihasa data-set [1], we were not able to find preprocessed Hindi data-set for the same.

Data-sets: We've created three data-sets while attempting to build a processed data-set.

1. Sanskrit to English translated data-set from itihasa [1]
2. Hindi data-set processed using OCR from Git Press.
3. English to Hindi translated data-set built using various translators

The data set consists of 585 chapters stored in the dictionary with chapter names as keys. 585 chapters are retrieved from the 7 kaands, and all the chapters in their respective kaands are merged as a single entity, totalling 585 chapters. Each dictionary key corresponds to each sarg, referring to the chapter name.

Web Scraping and Data Preprocessing:

Initially we tried scraping the data-set from the Valmiki Ramayana online website using the selenium web-driver and chrome driver tool. The data set was on different pages on the website that we had to traverse. We scrapped the links from the website as needed and opened the appropriate links to the different chapters on a separate web-driver element. Hence, we used two web-driver elements to process it.

There is no standard preprocessed corpus available for the Ramayana in Hindi, so we downloaded the Hindi Ramayana from Gita Press in PDF format. However, there is an issue with encoding the Hindi language for PDF format. Our first task was to convert the PDF data for the Hindi language into a text format for easy processing. We've converted the PDF into a set of images and then processed the images using an OCR to extract the text from the images.

For extracting data from the Gita press PDF, our first task was to convert the existing PDF into a set of images split by their respective pages. To convert the PDF to images we've used free online tools and then processed the generated images using python tesseract-OCR library to generate the data-set in the text format.

Once the text file was generated, we had to process and arrange the data-set as per their respective kaand and sarg. The data set is a dictionary of all the sargs arranged in order. The data-set points to a total of 585 chapters, and each dictionary key points to a list of all the shlokas as given in the Ramayana.

We've also experimented with translating the existing itihasa data set [1] from English to Hindi using the translators python library. The results were not good since the translators didn't perform well enough to maintain the essence of the Ramayana literature.

After careful analysis, though time-consuming, we decided to use the OCR method of extracting the text from images and directly use the Ramayana from Gita Press PDF.

Event Classification:

Once the data set was complete, the next step in the execution was to determine the semantic and syntactic properties of the data set, which would eventually help us order the events in temporal order.

We've used the dependency parsing mechanism to determine and segregate the past, present and future events.The term Dependency Parsing (DP) refers to the process of examining the dependencies between the phrases of a sentence in order to determine its grammatical structure. A sentence is divided into many sections based mostly on this. The process is based on the assumption that there is a direct relationship between each linguistic unit in a sentence. These hyperlinks are called dependencies.

Dependency parsing aims at discovering the syntactic dependency tree Z of an input sentence X, where X is a sequence of words $X_1, \ldots, X_n$ with length N. A dummy root word X_0 is typically added at the beginning of the sentence. A dependency tree Z is a set of directed edges between words that form a directed tree structure rooted at X_0. Each edge points from a parent word (also called a head word) to a child word. [3]

To perform dependency parsing we've used Stanford's NLP python library Stanza [6].Running the dependency parser requires the TokenizeProcessor, MWTProcessor, POSProcessor, and LemmaProcessor. After all these processors have been run, each Sentence in the output would have been parsed into Universal Dependencies (version 2) structure, where the head index of each Word can be accessed by the property head, and the dependency relation between the words.

Using dependency parsing, we are able to determine the tense of a given event. We parse through each sarg for the entire Ramayana data set and segregate past, present, and future events and store it in a list.

Once we have the list of tense events segregated, we run the ranking function to calculate the temporal order of the events for a given query.

Ranking Events based on a given query:

We are using the BM25[8] ranking algorithm to rank the events based on a given query. BM25 is a bag-of-words retrieval function that ranks a set of documents based on the query terms appearing in each document, regardless of their proximity within the document. It is a family of scoring functions with slightly different components and parameters. One of the most prominent instantiations of the function is as follows.

Given a query Q, containing keywords $q_1, q_2, ...q_n$ the BM25 score of a document D is:

$$Score(D,Q) = \sum_{1}^{N} IDF(q_i).\frac{f(q_i,D).(k_1+1)}{f(q_i,D)+k_1.(1-b+b.\frac{|D|}{avgdl})}$$

where $f(q_i, D)$ is q_i's term frequency in the document D in words, and avgdl is the average document length in the text collection from which documents are drawn. k and b are hyper-parameters which are set to 2 and 0.75 respectively.

$IDF(q_i)$ is the inverse document weight of the query term q_i which is computed as follows:

$$IDF(q_i) = \ln(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5}) + 1$$

where N is the total number of documents in the collection, and $n(q_i)$ is the number of documents containing q_i

We've ranked the events based on the BM25 algorithm for the given input query. Here, we are passing one sentence of a particular chapter id C, and model BM-25 returns the chapter name and score related to that query. Furthermore, we pick the top M chapters and if they are found in the chapters that have id less than C, then we can say it has happened in the past somewhere.

Analysis using Knowledge Graph:

In knowledge representation and reasoning, knowledge graph is a knowledge base that uses a graph-structured data model or topology to integrate data. Knowledge graphs are often used to store interlinked descriptions of entities – objects, events, situations or abstract concepts – while also encoding the semantics underlying the used terminology [7]

We are using knowledge graph to show the dependency and the relationships between various events across past, present and the future for better understanding. We use the subject and object retrieved from the dependency parser to figure out the relation between multiple nodes.

5 Results

Dependency parser output: From the below image we can observe that for a given event we are able to retrieve the tense of the event using dependency parser.

```
word: श्री       upos: PROPN     xpos: NNPC      feats: Case=Nom|Gender=Masc|Number=Sing|Person=3
word: राम        upos: PROPN     xpos: NNP       feats: Case=Acc|Gender=Masc|Number=Sing|Person=3
word: ने upos: ADP       xpos: PSP       feats: AdpType=Post
word: रावण       upos: PROPN     xpos: NNP       feats: Case=Acc|Gender=Masc|Number=Sing|Person=3
word: का         upos: ADP       xpos: PSP       feats: AdpType=Post|Case=Nom|Gender=Masc|Number=Sing
word: वध         upos: NOUN      xpos: NN        feats: Case=Nom|Gender=Masc|Number=Sing|Person=3
word: किया       upos: VERB      xpos: VM        feats: Aspect=Perf|Gender=Masc|Number=Sing|VerbForm=Part|Voice=Act
word: था         upos: AUX       xpos: VAUX      feats: Gender=Masc|Mood=Ind|Number=Sing|Tense=Past|VerbForm=Fin
```

Temporal ordering of events:

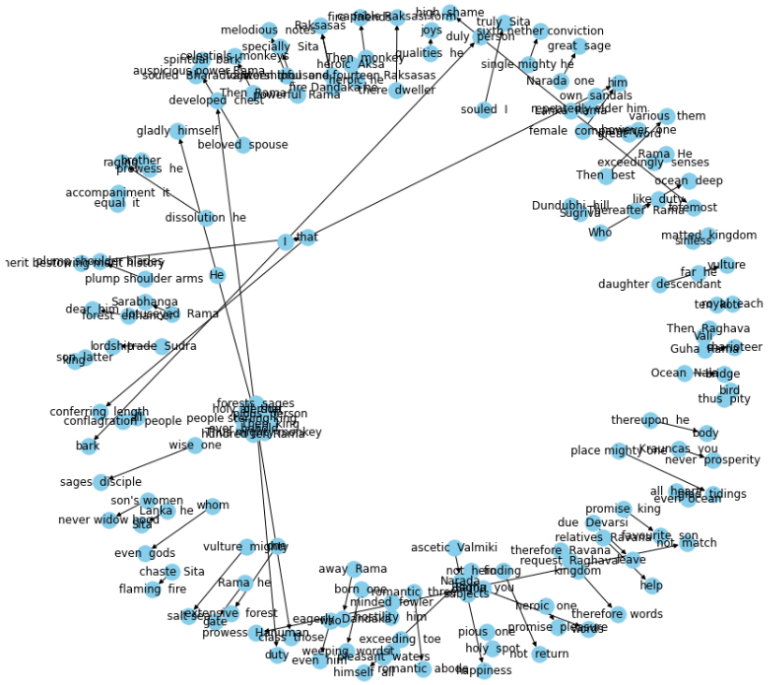
For a given query we rank the events based on BM25 algorithm and return top K events which have happened in the past. Present and future events are stored in the list in a separate file.

```
In [68]: k, b = 2, 0.75
         bm_dict = []
         BM_25("रीछ, वानर, लंगूर और बिलाव आदि जन्तु वहाँ निवास करते थे।")

          [(348, -108.56511039280001),
           (279, -112.35778264578785),
           (347, -126.31994385019546),
           (303, -126.97343900187376),
           (173, -128.21994904277793),
           (243, -129.23811623062826),
           (172, -129.31050826766472),
           (409, -131.39115954167735),
```

Knowledge Graph:

In the below knowledge graph each input node is a subject node and the output node is an object node, with the label mentioning the relationship between them. Using the below graph we can analyse the temporal relationship between the past, present and the future events.



6 Future Work

We did not use any meta data or external resources to solve the problem; we only used the raw data set, which does not account for the approach of using meta data.

The addition of meta data provides additional information about the occurrence of an event that was not present in the raw data set. The approach of adding meta data to the data set, such as the year of a particular popular event, and then performing temporal ordering using those timed events is yet to be explored.

7 Conclusion

Without specific temporal information such as time, date, duration, and set, there has been very little progress in the temporal ordering of events in ancient historical literature. This is a relatively new and difficult research area.

We demonstrated how to retrieve the temporal order of multiple correlated events using the dependency parsing technique, tagging the parts of speech, and knowledge graphs. This will serve as a good comprehensive approach to the problem statement in the absence of good temporal information.

References

[1] Aralikatte, R., de Lhoneux, M., Kunchukuttan, A., and Søgaard, A. Itihasa: A large-scale corpus for Sanskrit to English translation. In *Proceedings of the 8th Workshop on Asian Translation (WAT2021)* (Online, Aug. 2021), Association for Computational Linguistics, pp. 191–197.

[2] Goel, P., Prabhu, S., Debnath, A., Modi, P., and Shrivastava, M. Hindi TimeBank: An ISO-TimeML annotated reference corpus. In *16th Joint ACL - ISO Workshop on Interoperable Semantic Annotation PROCEEDINGS* (Marseille, May 2020), European Language Resources Association, pp. 13–21.

[3] Han, W., Jiang, Y., Ng, H. T., and Tu, K. A survey of unsupervised dependency parsing, 2020.

[4] Kanhabua, N., and Anand, A. Temporal information retrieval. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval* (New York, NY, USA, 2016), SIGIR '16, Association for Computing Machinery, p. 1235–1238.

[5] Pustejovsky, J., Castaño, J., Ingria, R., Saurí, R., Gaizauskas, R., Setzer, A., and Katz, G. Timeml: A specification language for temporal and event expressions.

[6] Qi, P., Zhang, Y., Zhang, Y., Bolton, J., and Manning, C. D. Stanza: A python natural language processing toolkit for many human languages, 2020.

[7] Wikipedia contributors. Knowledge graph — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Knowledge_graph&oldid=1082506545, 2022. [Online; accessed 26-April-2022].

[8] Wikipedia contributors. Okapi bm25 — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Okapi_BM25&oldid=1082128332, 2022. [Online; accessed 26-April-2022].

[9] Wikipedia contributors. Ramayana — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Ramayana&oldid=1084753316, 2022. [Online; accessed 26-April-2022].