Detection and normalisation of Temporal Expressions in Hindi language

Himani Kapur[1], Ashish Girdhar[2], Dr. Manoj Sachan[3]

[1]M Tech Student, Computer Science and Engineering, SLIET Longowal, Sangrur, India
[2] Lecturer, Computer Science and Engineering, Thapar University, Patiala, India
[3]Associate Professor,SLIET Longowal

---***---

Abstract - Temporal expressions are those expressions which convey some kind of temporal information ,i.e. related to time. These expressions can indicate a point in time such as “tomorrow 12 p.m.” or a period of time, e.g. “for first 7 months”. The task of recognizing temporal expressions from a chunk of text detects the temporal expressions and interprets them. Hence, it essentially consists of two sub tasks of detecting the temporal expressions and normalizing(interpreting) the temporal expressions. Interpretation of the temporal expressions is done in order to make them understandable to the computer algorithms. For Hindi language, the task of recognition has been achieved to some level but the task of interpretation of the detected temporal expression is still in progress. This paper attempts to achieve both detection and normalization of temporal expression in texts written in Hindi language with approximately 78% accuracy.

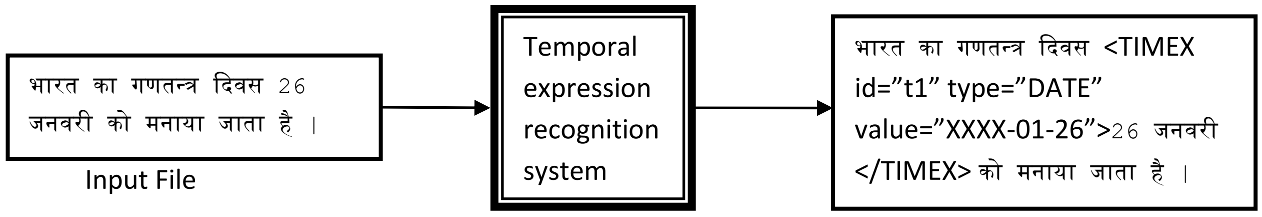
Key Words: Annotations, GATE tool, Normalization, Rule-based approach, temporal expressions

1. INTRODUCTION

Temporal expression recognition is defined as detecting the temporal expressions (i.e. words or numerals or a combination of both which indicate time) from the text and interpreting their values(normalization), which helps the computer algorithms to fetch the temporal information from the text easily. Temporal expression recognition results in annotating these expressions from the text or surrounding these expressions with <TIMEX/> tags. As depicted in the Fig.1, the output of the temporal annotation system is in the form of surrounding of temporal information with TIMEX tags. As observed by [1], the annotation of temporal expressions or Timex’s may act as a pre processing step for the temporal information retrieval task (T-IR). Since the temporal information present in the chunk of text can be identified by the recognition system and then the retrieval system may retrieve the relevant information from it. Hence the applicability of timex recognition is of importance. Earlier, timex recognition was considered a sub task of Named Entity Recognition and these were collectively identified as enamex, i.e. entity type. Later in MUC-5 explicit type of temporal expressions were first observed by [2] and the task of recognition of timex entities was studied. The corpora used for timex recognition purpose consists of news articles because of their extensive use of temporal expressions. Much of the work has been done in this area for many languages such as English (HeidelTime[3], DANTE[4], SUTime[5]) , Chinese (SUTime), etc. but for Hindi language authors could find only [6]. In this paper, we discuss the temporal expressions in detail and a rule-based approach for recognition of temporal expressions from the texts of Hindi language. The organization of the paper is done as follows: Section2 discusses the temporal expressions and their types in detail while in Section3 the proposed approach for recognizing the temporal entities in Hindi text is elaborated and the various issues have been encountered. Section 4 displays the results
obtained on the corpus used and in Section 5 we conclude the paper.

2. TEMPORAL ENTITIES AND THEIR TYPES

Temporal expressions are a rich form of natural language that can be defined as a sequence of tokens with temporal meaning.[1] Temporal expressions consist of linguistic expressions and numerals and have to be detected and



भारत का गणतन्त्र दिवस 26 जनवरी को मनाया जाता है ।

भारत का गणतन्त्र दिवस <TIMEX id=”t1” type=”DATE” value=”XXXX-01-26”>26 जनवरी </TIMEX> को मनाया जाता है ।

Fig 1. Annotation of temporal expression in Hindi text

recognized in order to aid the Temporal-Information Retrieval (T-IR) Because of the involvement of the linguistic words within the temporal entities, the complexity of the detection and normalization rises to greater levels.
As pointed by TIMEX2 community in [7], the temporal expressions can be broadly divided into two types, on the basis of duration on the timeline:

2.1 Point referring expressions

These are those expressions which indicate a particular point or an instant on the timeline. These expressions have zero duration of time. For example, following expressions in Fig.2 written in bold and italics are point referring expressions in Hindi.

a. प्रधानमंत्री अपनी जापान की यात्रा के लिए ***बुधवार*** को रवाना हुए ।
b. अविनाश का जन्म ***15 जनवरी*** को हुआ ।
c. ***अगले हफ्ते*** दो नई ट्रेनों का उद्घाटन किया जाएगा ।

Fig. 2 Examples of Point referring expressions

2.2 Period referring expressions

These are those entities which indicate a period of some duration on the timeline. These expressions occupy more space on the timeline than the point referring expressions. As shown in Fig.3, the words in written in bold and italics are period referring temporal entities in Hindi.

a. दीवाली का त्यौहार राजस्थान में ***5 दिनों*** तक मनाया जाता है ।
b. ***एक हफ्ते*** के बाद सैलरी बैंक खाते में आ जाएगी ।
c. महाभारत का युद्ध ***18 दिनों*** तक चला ।

Fig. 3 Examples of Period referring expressions

However, point and period referring expressions are further disambiguated into four different types of temporal entities- Date, Time, Duration and Set. Date and Time type of temporal expressions are, as indicative of their names, those entities which point to some date and time that is a single entity respectively. Duration and Set type of temporal entities are those expressions which convey some time period and an expression that is repetitive in nature, respectively. Examples of Date, Time, Duration and Set type of temporal entities are written Fig. 4 below.

The objective of this work is to develop system which could identify these expressions from the text and interpret their values. The first task is recognizing the temporal entities and the second task is normalization of the recognized entities. To the best of our knowledge, normalization of the temporal expressions in Hindi language has never been done before. This paper aims at achieving both detection and normalisation for Hindi language.

a. संसद पर हमला ***13 दिसम्बर*** को हुआ ।
b. संसद पर हमला ***13 दिसम्बर को दोपहर दो बजे*** हुआ ।
c. गणतन्त्र दिवस की परेड ***4 घण्टे*** की होती है ।
d. ***हर साल*** हम परेड देखने जाते हैं ।

Fig. 4 a. Date b. Time c. Duration and d. Set type of temporal expressions

3. STATE OF THE ART

There are two approaches broadly to the problem:
1. Rule- based approach
2. Statistical methods based approach
In rule-based approach, knowledge about the problem domain is brought to use for writing rules, or patterns which if match the particular matching rule gets fired and action is taken. Rule-based approach is easy to implement but requires the rule-base to be exhaustive so that each and every case has been taken care of. Nevertheless, rule-based approach suffers from a major drawback that it is difficult to port the system for other languages. This problem of rule-based approach in the area of temporal expression recognition has been dealt by [7] by doing an automatic translation of rules from Spanish to English, and then from both Spanish and English to Italian. Although the results could not yield a highly accurate system but it does provide some basis for start developing rules.

Machine learning approach or the statistical methods based approach uses the discriminative models such as CRF or SVM or may use the generative models such as HMM. Classifiers based on machine learning approach use features such as capitalization, containing digits, etc.

It has been argued that for recognition of temporal expressions statistical machine learning methods can be made to work but normalization of the entities after recognizing them, requires rules to be written [8]. So for normalizer to work, rules will have to be written by the knowledge expert.

The system proposed, works on the rule based approach for both recognizer and normaliser. Rules used for recognizer are regular expressions that match with the words of the sentence. The system developed for achieving these tasks consists of two sub systems- Recogniser and Normaliser, as shown in Fig.5. Recogniser is responsible for extracting the temporal entities from the chunk of text while Normaliser takes care of the interpreting the recognized expressions. Sometimes a reference time is also needed to interpret the information correctly, for which the normaliser looks up to the system clock on which it runs. The figure below gives the block diagram of the system. The normaliser deals with the matched text while the recognizer looks over the entire text.

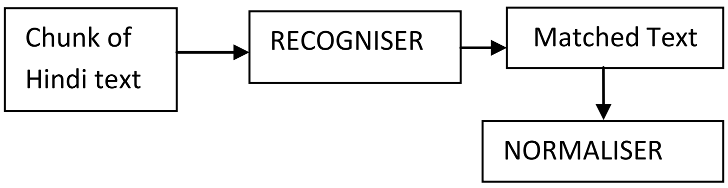


Fig. 5 Broad view of the system

3.1 Recogniser and Normaliser

Recognizer receives from the text, (|) delimited sentences which in Hindi language is considered as a full-stop. Since the target corpus consists of news articles only hence the delimiter is chosen as full stop only and question-mark(?) is not considered.(Although some sentences in other literary works may end in a question mark, but news articles are only informative in nature, hence question-mark is not taken as a delimiter). Rules are written using Regex package of Java language[9]. In order to match date, simple regular expressions can be written such as ([0-3][0-9])[-/.\\s](जनवरी|फरवरी|मार्च|अप्रैल|मई|जून|जुलाई|अगस्त|सितम्बर|अक्तूबर|नवम्बर|दिसम्बर)[-/.\\s][0-9]{4}) (where (\\s) indicates whitespace character) which can be used to match expressions like 18 जनवरी 2014 or 18-जनवरी-2014, etc. While writing rules, we came across the idea of obtaining Cartesian product between two rules to develop new rule as explored in [6]. So instead of writing separate rules for expressions like सोमवार रात, बुधवार रात, मंगलवार सुबह, शनिवार सुबह, Cartesian cross product has been taken for separate rules like (सोमवार|मंगलवार|बुधवार...) and (सुबह|रात|दोपहर|शाम) resulting in the desired rule. Such combinations were identified and hence the effort in writing rules got minimized. There are about 30 rules at present in the rule-base which combine based on the Cartesian product described above and hence make the rule-base more efficient. However since the corpus used for the present work consists of news article only hence the rules are written keeping that in mind. This implies that the rules written might not perform well in case of other sources such as literary works, etc.

Recogniser runs the entire chunk of text against all the regular expressions written for Date, Time, Duration and Set type of expressions. Once a match is found in the text, the matched text is picked from the text along with the type of regular expression with which it was matched. The contents of the chunk till the matching text are placed intact in the output while the matching text and the regular expression with which it matched are sent to the normaliser so that TIMEX tags can be surrounded to it.

Recogniser sends the matched text and the regular expression with which it matched to the Normaliser. Normaliser is responsible for assigning the type, i.e. whether a Date, Time, Duration or a Set type of temporal expression has been found.

In addition, Normaliser interprets the value of the matching text and hence is responsible for assigning the value to the <TIMEX> tag. Since both recognizer and normaliser are rule-based, recognizer helps the normaliser in providing a clue about the expression by sending the category with which match was found, along with the matched text.

Various issues cropped up while writing rules for recognition and normalization of temporal entities for Hindi language, of which some are put as under:

1. Map the date on the calendar:

Some words such as "रविवार", "सोमवार", "पिछले साल", etc have to be mapped to the correct date, in case of first two and correct year in case of last. For this, the system date is considered as the reference date and the mappings are done accordingly. Since Java supports lots of APIs, hence for achieving accurate mapping, joda-time API is used.[10].

2. Conversion of Hindi numerals:

Sometimes the expressions may consist of Hindi numerals such as "१६/१२/१९८९", "१५- जनवरी", etc which needs to be converted to "16/12/1989" and "15-January" respectively. Regular expression for English numerals was written separately and for Hindi numerals a separate expression was there. As soon as a Hindi numeral is detected in the matched text, it is replaced by its corresponding English numeral.

3. Calculation of number of days between two dates

Duration types of temporal entities usually have two dates, i.e. generally these type of expressions are written as "14 मई से 17 जून" in place of writing "35 दिन". So for these type of entities, calculating the number of days in between these two dates, joda-api was used.

4. Conversion of Hindi number in words to digits

Sometimes, the entities contain words only such as "दोपहर एक बजे". While encountering these entities, the Hindi words were first replaced by the English counterparts and then the interpretation was done.

5. Encountering plurals:

Hindi language has words which are same for singular and plural form such as "माह". But like all other Indian languages, Hindi is also vast and there exists more than one plural form of some words such as “महीने”,”महीनों” for singular word “महीना”, “हफ्ते”,“हफ्तों” for word “हफ्ता” and many such cases. All such cases have to be handled as well while hand-crafting rule-base.

6. Different words for one word used same time in document:

Hindi language has a vast vocabulary and there may exist more than one word with same meaning. For instance, “सप्ताह” and “हफ्ता” both imply week, likewise “शाम” and “सांय” both words mean evening in English. Also, “बृहस्पतिवार”, “वीरवार” and “गुरुवार” all convey Thursday in English.

7. Different spellings for same word:

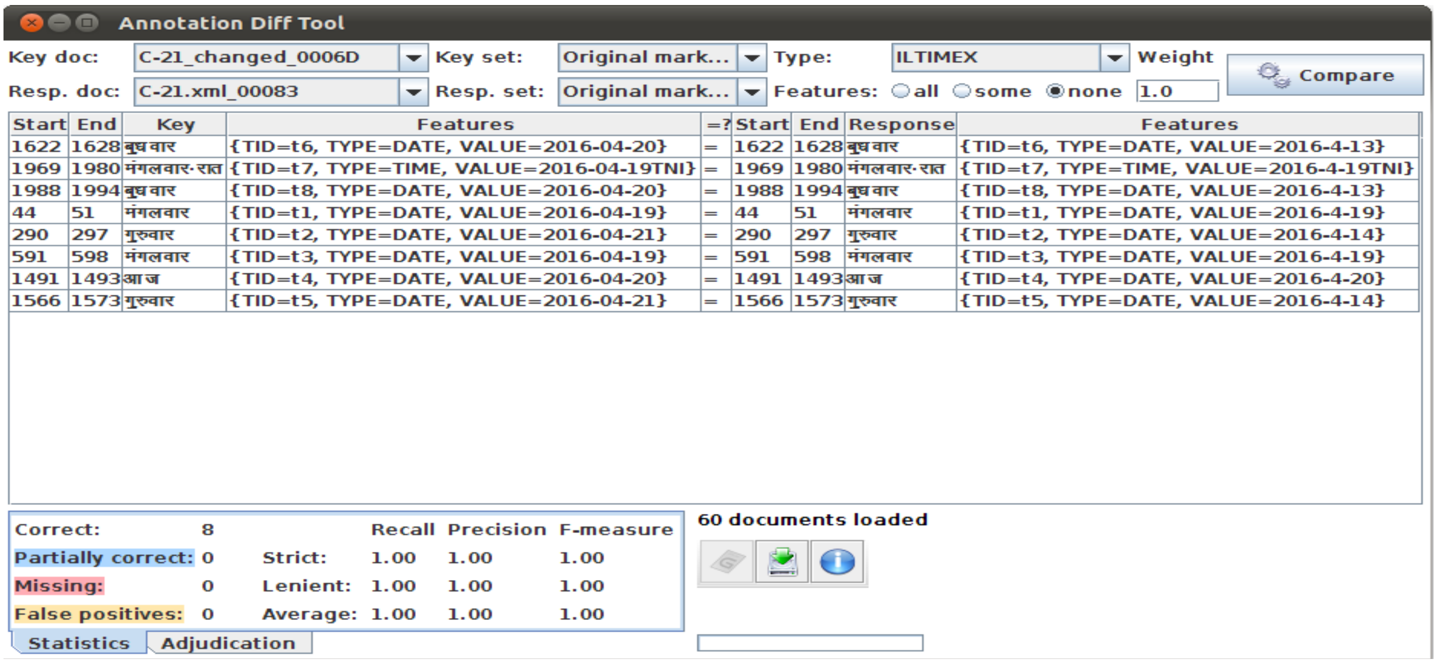
Hindi differs from English in many ways. In English language, within one dialect of English more than one spelling of a word is not allowed. But in Hindi, more than one spelling exist. For instance, “नवम्बर” and “नवंबर” both are correct. There may be possibility of presence of both different spellings in the same paragraph. The rule-base should be made exhaustively covering all such cases.

3.2 Binding of XML Elements and Java classes

Both the input and output files to be manipulated are written in XML. Input file was easily parsed but in the output file XML tags were to encoded. In order to deal with the mixed content(tags and text) of the XML Element, Java Architecture for XML Binding(JAXB)[11] was used which treats the JAXB-annotated Java classes as XML Elements. In order to generate JAXB-annotated classes using JAXB API, XSD schema of the XML file was required. XSD schema was generated making use of trang API [12].

3.3 Corpus used

The original corpus used for this work has been taken from[6]. The original corpus contained 300 news articles both as plain text and manually tagged. This corpus is taken from 300 plain documents of FIRE 2011[13]. The manually tagged articles are used as the gold standard for measuring the precision and recall of the system developed. However, the glitch in the manually tagged documents remains. The tagged temporal entities have not been normalized. So these files were updated to include normalized value. The current work hence is restricted to 30 updated documents only. More documents will be included when the work will be extended.


Annotation Diff Tool

Key doc: C-21_changed_0006D Key set: Original mark... Type: ILTIMEX Weight
Resp. doc: C-21.xml_00083 Resp. set: Original mark... Features: all some [✓ none] 1.0 Compare

Start	End	Key	Features	=?	Start	End	Response	Features
1622	1628	बुधवार	{TID=t6, TYPE=DATE, VALUE=2016-04-20}	=	1622	1628	बुधवार	{TID=t6, TYPE=DATE, VALUE=2016-4-13}
1969	1980	मंगलवार·रात	{TID=t7, TYPE=TIME, VALUE=2016-04-19TNI}	=	1969	1980	मंगलवार·रात	{TID=t7, TYPE=TIME, VALUE=2016-4-19TNI}
1988	1994	बुधवार	{TID=t8, TYPE=DATE, VALUE=2016-04-20}	=	1988	1994	बुधवार	{TID=t8, TYPE=DATE, VALUE=2016-4-13}
44	51	मंगलवार	{TID=t1, TYPE=DATE, VALUE=2016-04-19}	=	44	51	मंगलवार	{TID=t1, TYPE=DATE, VALUE=2016-4-19}
290	297	गुरुवार	{TID=t2, TYPE=DATE, VALUE=2016-04-21}	=	290	297	गुरुवार	{TID=t2, TYPE=DATE, VALUE=2016-4-14}
591	598	मंगलवार	{TID=t3, TYPE=DATE, VALUE=2016-04-19}	=	591	598	मंगलवार	{TID=t3, TYPE=DATE, VALUE=2016-4-19}
1491	1493	आज	{TID=t4, TYPE=DATE, VALUE=2016-04-20}	=	1491	1493	आज	{TID=t4, TYPE=DATE, VALUE=2016-4-20}
1566	1573	गुरुवार	{TID=t5, TYPE=DATE, VALUE=2016-04-21}	=	1566	1573	गुरुवार	{TID=t5, TYPE=DATE, VALUE=2016-4-14}

			Recall	Precision	F-measure
Correct:	8				
Partially correct:	0	Strict:	1.00	1.00	1.00
Missing:	0	Lenient:	1.00	1.00	1.00
False positives:	0	Average:	1.00	1.00	1.00

60 documents loaded

Fig. 6 Annotation Diff Tool for document C-21 of corpus

4. RESULTS

The results were obtained using the Annotation Diff Tool of the GATE developer tool 8.0 [14].Precision, recall and F-measure values of each document were used to calculate the average F-measure score. The accuracy of the system came out to be approximately 78% with the current 30 rules in the rule-base. Adding more rules to the rule base would increase the accuracy. The tool provides option to compare the tags based on some particular attribute, i.e. TID, TYPE or VALUE. The results however have been obtained for the complete tag in the current work. The main focus of this work was interpretation of the detected temporal entity and hence the complete tag of the key and response documents is compared. In Fig. 6 the Annotation diff tool shows the difference between the ILTIMEX(TIMEX tag for Indian Language) tags of the gold standard(manually tagged) document and the proposed system tagged document. Key document contains the manually tagged document while the Response document is the system annotated document. Any difference between the attributes or the value of the tag in the response document, is treated as a partially correct response by the Annotation diff tool of the GATE developer tool.

5. CONCLUSION

The proposed system shows about 78% accuracy on corpus containing news articles. The results drawn indicate that with the expansion of rules in rule-base, the system is expected to show better results. The normalization and detection of the temporal entities is done taking into consideration the fact that the corpus used contains only news articles. Thus, it can be prospected that the system might not be able to reproduce these results again when used for some other domain such as some literary work. Machine learning methods can also be used for detection purpose and hence the burden of using the rule-base for both detection and normalization would come down. Nevertheless, the proposed rule-based approach implemented in Java language is able to produce a 78% accurate system(as measured by GATE tool) which detects and interprets temporal entities in Hindi language.

REFERENCES

[1] Ricardo Campos, G Dias, A M. Jorge, and Adam Jatowt. 2014. Survey of temporal information retrieval and related applications. ACM Computing Survey Vol. 47, No. 2, Article 15 (July 2014), 41 pages.

[2] A. Setzer and R. J. Gaizauskas. 2000. Annotating events and temporal information in newswire texts. In Proceedings of the LREC'00

[3] Jannik Strotgen and Michael Gertz. Heideltime: High quality rule-based extraction and normalization of temporal expressions. In Proceedings of the 5th International Workshop on Semantic Evaluation, pages 321-324. Association for Computational Linguistics, 2010.

[4] Paweł Mazur and Robert Dale, A Rule Based Approach to Temporal Expression Tagging, Proceedings of the International Multiconference on Computer Science and Information Technology pp. 293–303,2007

[5] Angel X Chang and Christopher D Manning. Sutime: A library for recognizing and normalizing time expressions. In LREC, pages 3735-3740, 2012.

[6] Ramrakhiyani, N. and Majumder, P. 2015. Approaches to temporal expression recognition in Hindi. ACM Trans. Asian Low-Resour. Lang. Inf. Process.Vol. 14, No. 1, Article 2 (January 2015), 22 pages.

[7] E. Saquete, P. Martinez-Barco, R. Munoz, M. Negri, M. Speranza, and R. Sprugnoli.2006. Multilingual extension of a temporal expression normalizer using annotated corpora. In Proceedings of the EACL 2006 Workshop on Cross-Language Knowledge Induction, pages 1-8, Trento, Italy, April. Association for Computational Linguistics.

[8] H. Llorens, L. Derczynski, R. Gaizauskas, E. Saquete, TIMEN: An Open Temporal Expression Normalisation Resource. In Proceedings of the eigth international conference on language resources and evaluation(LREC 2012); 2012

[9] https://docs.oracle.com/javase/7/docs/api/java/util/regex/package-summary.html

[10] www.joda.org/joda-time

[11] https://jaxb.java.net

[12] www.thaiopensource.com/download/old/relaxng/20030122/trang-manual.html

[13] Palchowdhury, S., Majumder, P., Pal, D., Bandyopadhyay, A., and Mitra, M. 2013. Overview of FIRE 2011. In Multilingual Information Access in South Asian Languages, Springer, 1–12.

[14] https://gate.ac.uk/download

[15] F. Schilder and C. Habel. 2001. From temporal expressions to temporal information: Semantic tagging of news messages. In L. Harper, I. Mani, and B. Sundheim, editors, Proceedings of the ACL 2001 Workshop on Temporal and Spatial Information Processing, pages 65-72, Toulouse, France, July. Association for Computational Linguistics.

[16] I. Mani and G. Wilson. 2000. Robust temporal processing of news. In Proceedings of the 38th Annual Meeting on Association for Computational Linguistics (ACL), pages 69-76, Hong Kong, October. Association for Computational Linguistics.