Time Expression Analysis and Recognition Using Syntactic Token Types and General Heuristic Rules

Xiaoshi Zhong and Erik Cambria
Computer Science and Engineering
Nanyang Technological University
{xszhong, cambria}@ntu.edu.sg
Outline

• **Time expression analysis**
  • Datasets: TimeBank, Gigaword, WikiWars, Tweets
  • Findings: short expression, occurrence, small vocabulary, similar syntactic behavior

• **Time expression recognition**
  • SynTime: syntactic token types and general heuristic rules
  • Baselines: HeidelTime, SUTime, UWTime
Time Expression Analysis

• Datasets
  • TimeBank
  • Gigaword
  • WikiWars
  • Tweets

• Findings
  • Short expression
  • Occurrence
  • Small vocabulary
  • Similar syntactic behaviour

Example time expressions:
  • now
  • today
  • Friday
  • February
  • the last week
  • 13 January 1951
  • June 30, 1990
  • 8 to 20 days
  • the third quarter of 1984
  • …
Time Expression Analysis - Datasets

•Datasets
  • TimeBank: a benchmark dataset used in TempEval series
  • Gigaword: a large dataset with generated labels and used in TempEval-3
  • WikiWars: a specific domain dataset collected from Wikipedia about war
  • Tweets: a manually labeled dataset with informal text collected from Twitter

•Statistics of the datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Docs</th>
<th>#Words</th>
<th>#TIMEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>TimeBank</td>
<td>183</td>
<td>61,418</td>
<td>1,243</td>
</tr>
<tr>
<td>Gigaword</td>
<td>2,452</td>
<td>666,309</td>
<td>12,739</td>
</tr>
<tr>
<td>WikiWars</td>
<td>22</td>
<td>119,468</td>
<td>2,671</td>
</tr>
<tr>
<td>Tweets</td>
<td>942</td>
<td>18,199</td>
<td>1,127</td>
</tr>
</tbody>
</table>

The four datasets vary in source, size, domain, and text type, but we will see that their time expressions demonstrate similar characteristics.
Time Expression Analysis – Finding 1

• **Short expression**: time expressions are very short.

80% of time expressions contain $\leq 3$ words
90% of time expressions contain $\leq 4$ words

Average length of time expressions

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Average length</th>
</tr>
</thead>
<tbody>
<tr>
<td>TimeBank</td>
<td>2.00</td>
</tr>
<tr>
<td>Gigaword</td>
<td>1.70</td>
</tr>
<tr>
<td>WikiWars</td>
<td>2.38</td>
</tr>
<tr>
<td>Tweets</td>
<td>1.51</td>
</tr>
</tbody>
</table>

Average length: about 2 words

Time expressions follow a similar length distribution
Time Expression Analysis – Finding 2

**Occurrence:** most of time expressions contain time token(s).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>TimeBank</td>
<td>94.61</td>
</tr>
<tr>
<td>Gigaword</td>
<td>96.44</td>
</tr>
<tr>
<td>WikiWars</td>
<td>91.81</td>
</tr>
<tr>
<td>Tweets</td>
<td>96.01</td>
</tr>
</tbody>
</table>

Example time tokens (red):

- now
- today
- Friday
- February
- the last week
- 13 January 1951
- June 30, 1990
- 8 to 20 days
- the third quarter of 1984
...
Time Expression Analysis – Finding 3

- **Small vocabulary**: only a small group of time words are used to express time information.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Words</th>
<th>#Time tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>TimeBank</td>
<td>130</td>
<td>64</td>
</tr>
<tr>
<td>Gigaword</td>
<td>214</td>
<td>80</td>
</tr>
<tr>
<td>WikiWars</td>
<td>224</td>
<td>74</td>
</tr>
<tr>
<td>Tweets</td>
<td>107</td>
<td>64</td>
</tr>
</tbody>
</table>

45 distinct time tokens appear in all the four datasets. That means, time expressions highly overlap at their time tokens.

<table>
<thead>
<tr>
<th>#Words</th>
<th>#Time tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>350</td>
<td>123</td>
</tr>
</tbody>
</table>

Overlap at year.
Time Expression Analysis – Finding 4

• **Similar syntactic behavior**: (1) POS information cannot distinguish time expressions from common text, but (2) within time expressions, POS tags can help distinguish their constituents.
  
  • (1) For the top 40 POS tags (10 × 4 datasets), 37 have percentage lower than 20%, other 3 are CD.
  
  • (2) Time tokens mainly have NN* and RB, modifiers have JJ and RB, and numerals have CD.
Time Expression Analysis – Eureka!

• **Similar syntactic behavior**: (1) POS information cannot distinguish time expressions from common text, but (2) within time expressions, POS tags can help distinguish their constituents.
  - (1) For the top 40 POS tags (10 × 4 datasets), 37 have percentage lower than 20%, other 3 are CD.
  - (2) Time tokens mainly have NN* and RB, modifiers have JJ and RB, and numerals have CD.

*When seeing (2), we realize that this is exactly how linguists define part-of-speech for language; similar words have similar syntactic behaviour. The definition of part-of-speech for language inspires us to define a type system for the time expression, part of language.*

Our Eureka! moment
Time Expression Analysis - Summary

• Summary
  • On average, a time expression contains two tokens; one is time token and the other is modifier/numeral. And the time tokens are in small size.

• Idea for recognition
  • To recognize a time expression, we first recognize the time token, then recognize the modifier/numeral.
Time Expression Analysis - Idea

• Summary
  • On average, a time expression contains two tokens; one is time token and the other is modifier/numeral. And the time tokens are in small size.

• Idea for recognition
  • To recognize a time expression, we first recognize the time token, then recognize the modifier/numeral.

20 days; this week; next year; July 29; …
Time Expression Analysis - Idea

• Summary
  • On average, a time expression contains two tokens; one is time token and the other is modifier/numeral. And the time tokens are in small size.

• Idea for recognition
  • To recognize a time expression, we first recognize the time token, then recognize the modifier/numeral.

Time token

20 days; this week; next year; July 29; …
Time Expression Analysis - Idea

• Summary
  • On average, a time expression contains two tokens; one is time token and the other is modifier/numeral. And the time tokens are in small size.

• Idea for recognition
  • To recognize a time expression, we first recognize the time token, then recognize the modifier/numeral.

Time token — Modifier/Numeral

20 days; this week; next year; July 29; …
Time Expression Recognition

• SynTime
  • Syntactic token types
  • General heuristic rules

• Baseline methods
  • HeidelTime
  • SUTime
  • UWTime

• Experiment datasets
  • TimeBank
  • WikiWars
  • Tweets
Time Expression Recognition - SynTime

• Syntactic token types
• General heuristic rules
Time Expression Recognition - SynTime

• Syntactic token types – A type system
  • Time token: explicitly express time information, e.g., “year”
    • 15 token types: DECADE, YEAR, SEASON, MONTH, WEEK, DATE, TIME, DAY_TIME, TIMELINE, HOLIDAY, PERIOD, DURATION, TIME_UNIT, TIME_ZONE, ERA
  • Modifier: modify time tokens, e.g., “next” modifies “year” in “next year”
    • 5 token types: PREFIX, SUFFIX, LINKAGE, COMMA, IN_ARTICLE
  • Numeral: ordinals and numbers, e.g., “10” in “next 10 years”
    • 1 token type: NUMERAL
  • **Token types to tokens is like POS tags to words**
    • POS tags: next/JJ 10/CD years/NNS
    • Token types: next/PREFIX 10/PREFIX NUMERAL years/TIME_UNIT
Time Expression Recognition - SynTime

• General heuristic rules
  • Only relevant to token types
  • Independent of specific tokens
<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Token level</td>
<td>time-related tokens and token regular expressions</td>
</tr>
<tr>
<td>Type level</td>
<td>token types group the tokens and token regular expressions</td>
</tr>
<tr>
<td>Rule level</td>
<td>heuristic rules work on token types and are independent of specific tokens</td>
</tr>
</tbody>
</table>

**Token level:** time-related tokens and token regular expressions

**Type level:** token types group the tokens and token regular expressions

**Rule level:** heuristic rules work on token types and are independent of specific tokens
SynTime – Overview in practice

- Add keywords under defined token types and do not change any rules
- Import token regex to time token, modifier, numeral
- Identify time tokens
- Identify modifiers and numerals by expanding the time tokens’ boundaries
- Extract time expressions
An example: the third quarter of 1984

A sequence of tokens: the third quarter of 1984
An example: the third quarter of 1984

Assign tokens with token types

A sequence of tokens:

```
PREFIX   NUMERAL   TIME_UNIT   PREFIX   YEAR
↑         ↑         ↑           ↑         ↑
the       third     quarter     of       1984
```
An example: the third quarter of 1984

Identify time tokens

Assign tokens with token types

A sequence of tokens:

Heuristic Rules

PREFIX NUMERAL TIME_UNIT PREFIX YEAR

the third quarter of 1984
An example: the third quarter of 1984

Identify modifiers and numerals by searching time tokens’ surroundings

Identify time tokens

Assign tokens with token types

A sequence of tokens:

Heuristic Rules

PREFIX  NUMERAL  TIME_UNIT  PREFIX  YEAR

the  third  quarter  of  1984
An example: the third quarter of 1984

Identify modifiers and numerals by searching time tokens’ surroundings.

Identify time tokens.

Assign tokens with token types.

A sequence of tokens:

The third quarter of 1984

Heuristic Rules

PREFIX NUMERAL TIME_UNIT PREFIX YEAR

the third quarter of 1984
An example: the third quarter of 1984

Identify modifiers and numerals by searching time tokens’ surroundings

Identify time tokens

Assign tokens with token types

A sequence of tokens:

Heuristic Rules

PREFIX NUMERAL TIME_UNIT PREFIX YEAR

the third quarter of 1984
An example: the third quarter of 1984

Identify **modifiers** and **numerals** by searching **time tokens’** surroundings

Identify **time tokens**

Assign tokens with token types

A sequence of tokens:
An example: the third quarter of 1984

Identify modifiers and numerals by searching time tokens’ surroundings.

Identify time tokens

Assign tokens with token types

A sequence of tokens:

Heuristic Rules

PREFIX NUMERAL TIME_UNIT PREFIX YEAR

the third quarter of 1984
An example: the third quarter of 1984

A sequence of token types

PREFIX NUMERAL TIME_UNIT PREFIX YEAR
An example: the third quarter of 1984

A sequence of token types

Export a sequence of tokens as time expression
An example: the third quarter of 1984
Time Expression Recognition - Experiments

• SynTime
  • SynTime-I: Initial version
  • SynTime-E: Expanded version, adding keywords to SynTime-I
    (Add keywords under the defined token types and do not change any rules.)

• Baseline methods
  • HeidelTime: rule-based method
  • SUTime: rule-based method
  • UWTime: learning-based method

• Experiment datasets
  • TimeBank: comprehensive data in formal text
  • WikiWars: specific domain data in formal text
  • Tweets: comprehensive data in informal text
Overall performance. The **best results** are in boldface and the **second best** are underlined. Some results are borrowed from their original papers and the papers indicated by the references.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Methods</th>
<th>Strict Match</th>
<th>Relexed Match</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Pr.</td>
<td>Re.</td>
</tr>
<tr>
<td>TimeBank</td>
<td>HeidelTime (Strotgen et al., 2013)</td>
<td>83.85</td>
<td>78.99</td>
</tr>
<tr>
<td></td>
<td>SUTime (Chang and Manning, 2013)</td>
<td>78.72</td>
<td>80.43</td>
</tr>
<tr>
<td></td>
<td>UWTime (Lee et al., 2014)</td>
<td>86.10</td>
<td>80.40</td>
</tr>
<tr>
<td></td>
<td>SynTime-I</td>
<td>91.43</td>
<td>92.75</td>
</tr>
<tr>
<td></td>
<td>SynTime-E</td>
<td><strong>91.49</strong></td>
<td><strong>93.48</strong></td>
</tr>
<tr>
<td>WikiWars</td>
<td>HeidelTime (Lee et al., 2014)</td>
<td>85.20</td>
<td>79.30</td>
</tr>
<tr>
<td></td>
<td>SUTime</td>
<td>78.61</td>
<td>76.69</td>
</tr>
<tr>
<td></td>
<td>UWTime (Lee et al., 2014)</td>
<td><strong>87.70</strong></td>
<td><strong>78.80</strong></td>
</tr>
<tr>
<td></td>
<td>SynTime-I</td>
<td>80.00</td>
<td>80.22</td>
</tr>
<tr>
<td></td>
<td>SynTime-E</td>
<td>79.18</td>
<td><strong>83.47</strong></td>
</tr>
<tr>
<td>Tweets</td>
<td>HeidelTime</td>
<td><strong>89.58</strong></td>
<td>72.88</td>
</tr>
<tr>
<td></td>
<td>SUTime</td>
<td>76.03</td>
<td>77.97</td>
</tr>
<tr>
<td></td>
<td>UWTime</td>
<td>88.54</td>
<td>72.03</td>
</tr>
<tr>
<td></td>
<td>SynTime-I</td>
<td>89.52</td>
<td>94.07</td>
</tr>
<tr>
<td></td>
<td>SynTime-E</td>
<td>89.20</td>
<td><strong>94.49</strong></td>
</tr>
</tbody>
</table>
## Difference from other Rule-based Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>SynTime</th>
<th>Other rule-based methods</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Layout</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rule level</td>
<td>General Heuristic Rules</td>
<td>Deterministic Rules</td>
</tr>
<tr>
<td>Type level</td>
<td>Time Token, Modifier, Numeral</td>
<td></td>
</tr>
<tr>
<td>Token level</td>
<td>1989, February, 12:55, this year, 3 months ago, ...</td>
<td>1989, February, 12:55, this year, 3 months ago, ...</td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>Heuristic rules work on token types and are independent of specific tokens, thus they are independent of specific domains and specific text types and specific languages.</td>
<td>Deterministic rules directly work on tokens and phrases in a fixed manner, thus the taggers lack flexibility</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td></td>
<td>/the/? [{tag:JJ}]? ($NUM_ORD) /-/? [{tag:JJ}]? /quarter/</td>
</tr>
</tbody>
</table>

Example: Heuristic Rules

- `PREFIX` to `NUMERAL` to `TIME_UNIT` to `PREFIX` to `YEAR`
A simple idea

Rules can be designed with generality and heuristics