Learning Text Patterns using Separate-and-Conquer GP

A. Bartoli, A. De Lorenzo, E. Medvet, F. Tarlao
University of Trieste, Italy
The Problem (I)

- Entity *extraction* from *unstructured* text
- **Syntactic** pattern
- Specified only by *examples* of desired (un)extractions

- Generate *regular expression* automatically
- Which “*generalizes*” the examples
The Problem (II)

- Multiple patterns possibly needed

18.12.2013
2007/01/09
23/03/2009
14-09-2011
23 July 2001
December 31, 2001
2000.01.27
Dec 31, 1991
1997/12/31
Regex learning by examples

- Long-standing problem

- Much research on classification

- Little research on extraction
Eric and Fabiano: During our month-end processes I have researched deal #549162.1. Could not find anything useful. Sorry, Pinco Pallo Executive Assistant to Ucio 713.853.5984 713.646.8381 (fax) pinco.pallo@malelab.it \DIA UniTS\ <info@malelab.it> on 12/04/2000
Regex learning by examples

● Long-standing problem

● Much research on classification

● Little research on extraction

● Hardly useful for “practical text processing”
  ○ Example: input string is a sequence of 20 symbols and symbols are bits
Our work in a nutshell: Interface

- **Input:**
  - Unstructured text file
  - Annotated with all the *desired extractions*

- **Output:**
  - Java/Javascript-compatible *regex*
  - Composed of *multiple* regexes “glued” by OR (“ | ”)
  - Each *capturing one pattern*
  - r1 | r2 | r3
Note

- **Input:**
  - Unstructured text file
  - Annotated with all the desired extractions

- No hints on patterns
  - How many
  - How they look like

- No hints on regexes

- Everything “discovered automatically”
Our work in a nutshell: Implementation

- GP-based system
  - Suitable for “practical problems”
  - “Much better” than earlier proposals

- Unable to cope with multipattern

- Modify & Extend for multipattern
Separate-and-Conquer():
Basic Idea (I)

● GP-Search() optimizes:
  ○ Extract only correct snippets (precision)
  ○ Extract all snippets that have to be extracted (recall)

● Tailor GP-Search() to:
  ○ Extract only correct snippets (precision)
  ○ Extract all snippets that have to be extracted (recall)
Separate-and-Conquer(): Basic Idea (II)

- GP-Search() generates regex:
  - Perfect precision
  - Misses extractions (non-perfect recall)

1. \( r := \text{GP-Search}(\text{Training}) \)
2. Remove from \( \text{Training} \) strings extracted by \( r \)
3. Repeat until \( \text{Training} \) is empty

4. Glue all \( r \) by OR
Separate-and-Conquer():
More details

resultSet := ∅;
Loop:
    regex := GP-Search(Training);
    if Precision(regex, Training) == 1
    then resultSet += regex;
    else exit-Loop;
    Training := Training - extractions(regex, Training);
    if Training == ∅ then exit-Loop
Glue resultSet by OR
GP-Search()

- Individual: regex (as a tree)
- Terminals: Training set-dependent (tokens)
- Initial population: Training set-dependent
  - Generalizations of desired extractions
  - Random
- Structural diversity
- Multiobjective fitness
  - Precision
  - Accuracy
  - Length (to be minimized)
- Multi-layered ranking
Full procedure

- Learning = Training + Validation

1. Execute $J$ Separate-and-Conquer(Training)

2. Compute F-measure of $J$ regexes on Learning

3. Choose regex with highest F-measure
Evaluation: Datasets

- Quite challenging
- Bills:
  - 600 portions of US Congress bills
  - ≈ 3000 Extractions: date in several formats
- Tweets:
  - 50000 tweets
  - ≈ 70000 Extractions: URLs, Hashtags, Twitter citations
- Headers:
  - 100 email headers (raw format)
  - ≈ 1500 Extractions: IP addresses, dates

- Bills available on our website
<table>
<thead>
<tr>
<th>Bills</th>
<th>Tweets</th>
<th>Headers</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.12.2013</td>
<td>@joshua_seaton</td>
<td>10.236.182.42</td>
</tr>
<tr>
<td>2007/01/09</td>
<td>#annoyed</td>
<td>Thu, 12 Jan 2012 04:33:34 -0800</td>
</tr>
<tr>
<td>23/03/2009</td>
<td><a href="http://t.co/Bw7A5sbI">http://t.co/Bw7A5sbI</a></td>
<td>93.174.66.112</td>
</tr>
<tr>
<td>14-09-2011</td>
<td>#Anonymous</td>
<td>209.85.216.53</td>
</tr>
<tr>
<td>23,July, 2001</td>
<td>@YourAnonNews</td>
<td>24 Jan 2011 09:36:00 -0000</td>
</tr>
<tr>
<td>December 31, 2001</td>
<td>@zataz</td>
<td>27 Apr 2011 09:31:01.0953</td>
</tr>
<tr>
<td>2000.01.27</td>
<td>@_SweetDiccWilly</td>
<td>Mon, 01 Oct 2012 12:05:40 +0000</td>
</tr>
<tr>
<td>Dec 31, 1991</td>
<td><a href="http://t.co/bYxJ9NAE">http://t.co/bYxJ9NAE</a></td>
<td>Mon, 01 Oct 2012 12:05:40 +0000</td>
</tr>
<tr>
<td>1997/12/31</td>
<td>#OpBlitzkrieg</td>
<td>151.76.78.168</td>
</tr>
</tbody>
</table>
Evaluation: Procedure

- For each dataset, 15 random tasks
  - 5 Training sets for each of 3 sizes
    (25, 50, 100 extractions)
- J = 32
  - 500 individuals, 1000 generations

Baseline: Computer

Automatic Synthesis of Regular Expressions from Examples

- "Much better" than earlier regex learning proposals
  (for text extraction)
## Key results: F-measure

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Num. of slices</th>
<th>Our method</th>
<th>Baseline</th>
<th>ΔFm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Fm</td>
<td>Fm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bills</td>
<td>25</td>
<td>0.49</td>
<td>0.24</td>
<td>104%</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.62</td>
<td>0.27</td>
<td>129%</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.73</td>
<td>0.39</td>
<td>87%</td>
</tr>
<tr>
<td>Tweets</td>
<td>25</td>
<td>0.94</td>
<td>0.87</td>
<td>8%</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.96</td>
<td>0.85</td>
<td>13%</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.99</td>
<td>0.90</td>
<td>10%</td>
</tr>
<tr>
<td>Headers</td>
<td>25</td>
<td>0.79</td>
<td>0.41</td>
<td>93%</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.90</td>
<td>0.44</td>
<td>104%</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.90</td>
<td>0.54</td>
<td>67%</td>
</tr>
</tbody>
</table>

- **F-measure**
  - Significant improvement
  - Absolute values “practically useful”
### Key results: Multipattern

- Effectively discovers different patterns
- ...without exaggerating
  - Targets would be 3 / 2 / 3

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Num. of slices</th>
<th>Our method</th>
<th>Baseline</th>
<th>ΔFm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Fm</td>
<td></td>
<td>Fm</td>
</tr>
<tr>
<td>Bills</td>
<td>25</td>
<td>0.49</td>
<td>3.2</td>
<td>0.24</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.62</td>
<td>4.0</td>
<td>0.27</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.73</td>
<td>4.6</td>
<td>0.39</td>
</tr>
<tr>
<td>Tweets</td>
<td>25</td>
<td>0.94</td>
<td>2.4</td>
<td>0.87</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.96</td>
<td>2.6</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.99</td>
<td>3.0</td>
<td>0.90</td>
</tr>
<tr>
<td>Headers</td>
<td>25</td>
<td>0.79</td>
<td>3.2</td>
<td>0.41</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.90</td>
<td>3.6</td>
<td>0.44</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.90</td>
<td>3.6</td>
<td>0.54</td>
</tr>
</tbody>
</table>
### Key results: Computational effort

- Less character evaluations \((10^{10})\)
- Usually (but not always) smaller execution time
  - tens of minutes

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Num. of slices</th>
<th>Our method</th>
<th>Baseline</th>
<th>ΔFm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Fm</td>
<td>CE</td>
<td>Fm</td>
</tr>
<tr>
<td>Bills</td>
<td>25</td>
<td>0.49</td>
<td>2.3</td>
<td>0.24</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.62</td>
<td>6.9</td>
<td>0.27</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.73</td>
<td>11.3</td>
<td>0.39</td>
</tr>
<tr>
<td>Tweets</td>
<td>25</td>
<td>0.94</td>
<td>0.6</td>
<td>0.87</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.96</td>
<td>1.6</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.99</td>
<td>3.2</td>
<td>0.90</td>
</tr>
<tr>
<td>Headers</td>
<td>25</td>
<td>0.79</td>
<td>4.6</td>
<td>0.41</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.90</td>
<td>7.6</td>
<td>0.44</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.90</td>
<td>15.1</td>
<td>0.54</td>
</tr>
</tbody>
</table>
Source code will be made public soon (GitHub)
Thanks for your attention

University of Trieste, Italy
http://machinelearning.inginf.units.it

@MaleLabT