# Learning Text Patterns using Separate-and-Conquer GP

A.Bartoli, A.De Lorenzo,
E.Medvet, F.Tarlao
*University of Trieste, Italy*

MACHINE LEARNING LAB

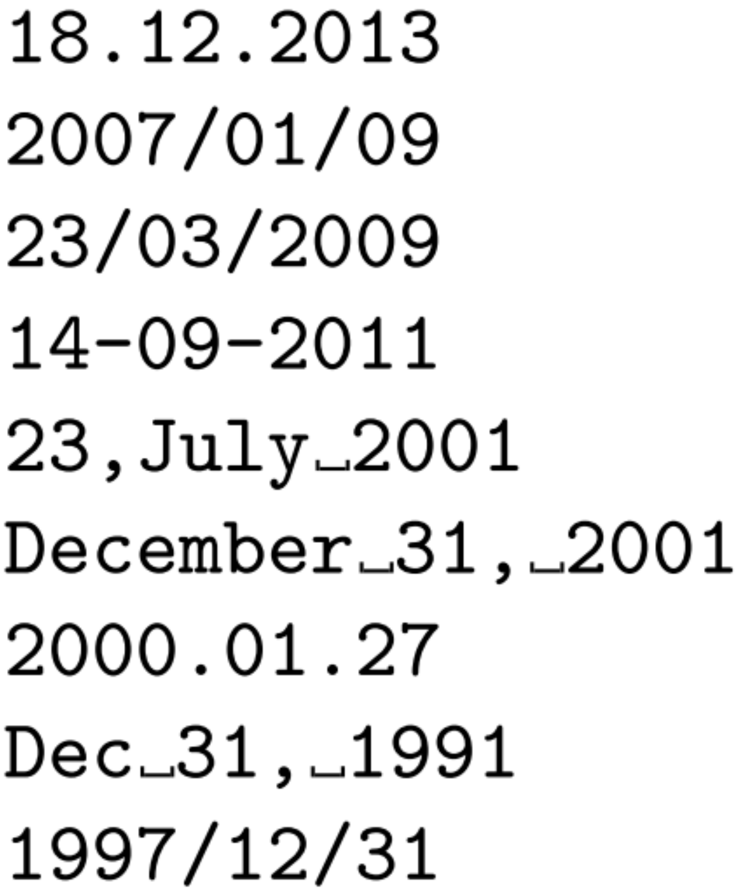
# The Problem (I)

- 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
```

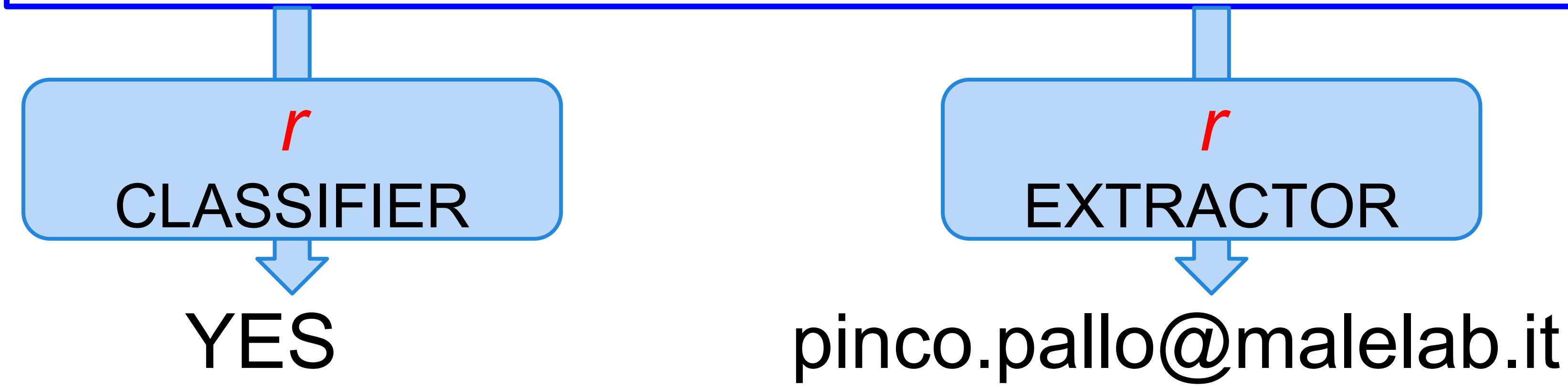
# Regex learning by examples

- Long-standing problem

- Much research on **classification**
- Little research on **extraction**

# Regex:
# Classification vs 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

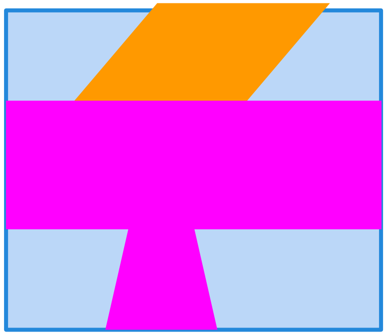
- Much research on **classification**
- Little research on **extraction**

- 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**

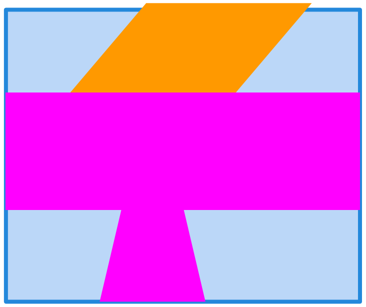
- Output:
  - Java/Javascript-compatible **regex**
  - Composed of **multiple** regexes “glued” by OR (“ | ”)
  - Each **capturing one pattern**
  - `r1 | r2 | r3`



# Note

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

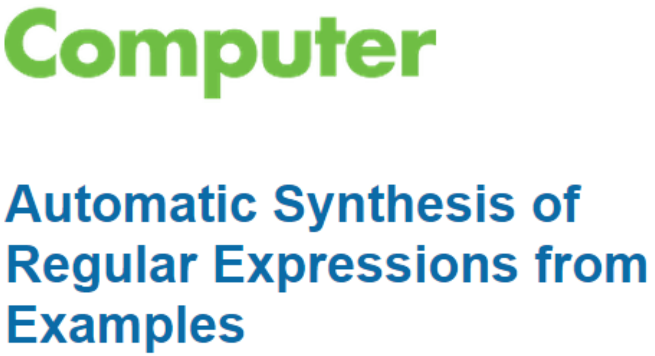
- No hints on patterns
  - How many
  - How they look like
- No hints on regexes



- **Everything “discovered automatically”**

# Our work in a nutshell: Implementation

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



Computer

Automatic Synthesis of Regular Expressions from Examples

- Unable to cope with **multipattern**

- Modify & Extend for multipattern

# Separate-and-Conquer(): Basic Idea (I)

Computer

Automatic Synthesis of Regular Expressions from Examples

- 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 := GP-Search(Training)
2. Remove from Training strings extracted by r
3. Repeat until 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

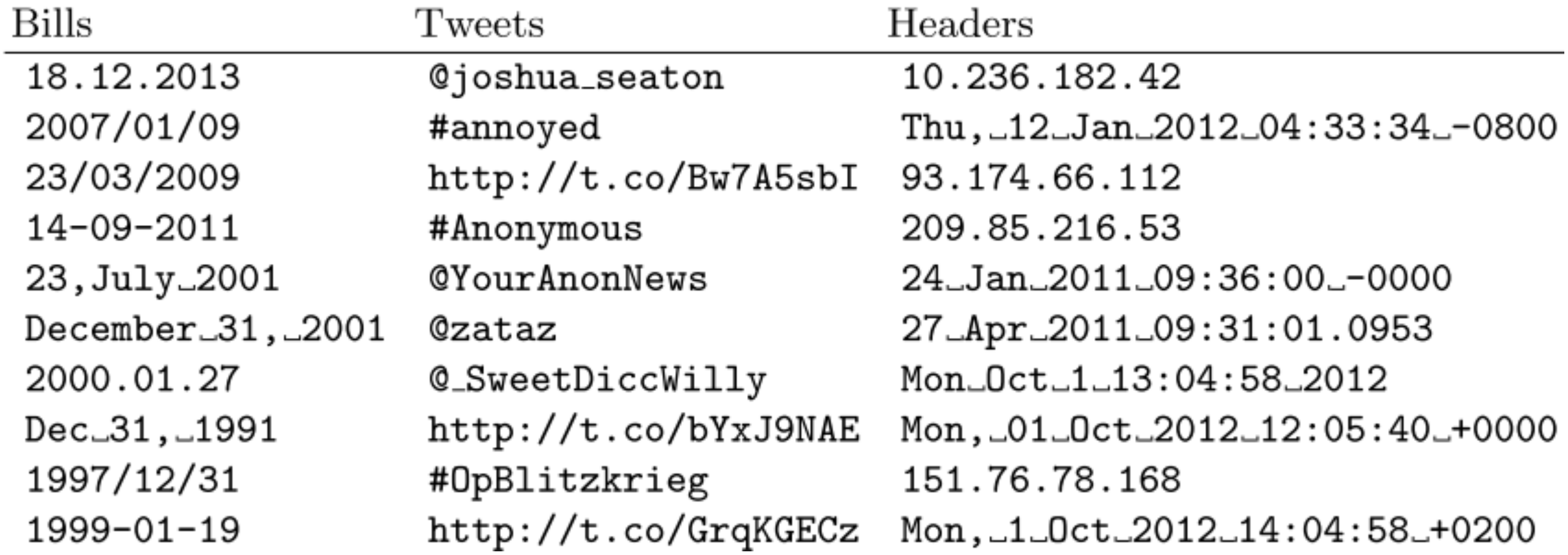
1. Execute ***J*** Separate-and-Conquer(Training)
2. Compute F-measure of ***J*** regexes on Learning
3. Choose regex with highest F-measure

# 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

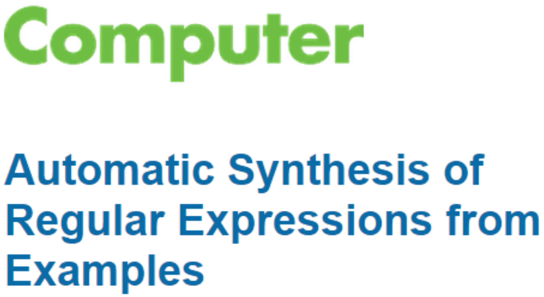
# A glimpse at extractions...



| Bills | Tweets | Headers |
|---|---|---|
| 18.12.2013 | @joshua_seaton | 10.236.182.42 |
| 2007/01/09 | #annoyed | Thu,␣12␣Jan␣2012␣04:33:34␣-0800 |
| 23/03/2009 | http://t.co/Bw7A5sbI | 93.174.66.112 |
| 14-09-2011 | #Anonymous | 209.85.216.53 |
| 23,July␣2001 | @YourAnonNews | 24␣Jan␣2011␣09:36:00␣-0000 |
| December␣31,␣2001 | @zataz | 27␣Apr␣2011␣09:31:01.0953 |
| 2000.01.27 | @_SweetDiccWilly | Mon␣Oct␣1␣13:04:58␣2012 |
| Dec␣31,␣1991 | http://t.co/bYxJ9NAE | Mon,␣01␣Oct␣2012␣12:05:40␣+0000 |
| 1997/12/31 | #OpBlitzkrieg | 151.76.78.168 |
| 1999-01-19 | http://t.co/GrqKGECz | Mon,␣1␣Oct␣2012␣14:04:58␣+0200 |

# 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:

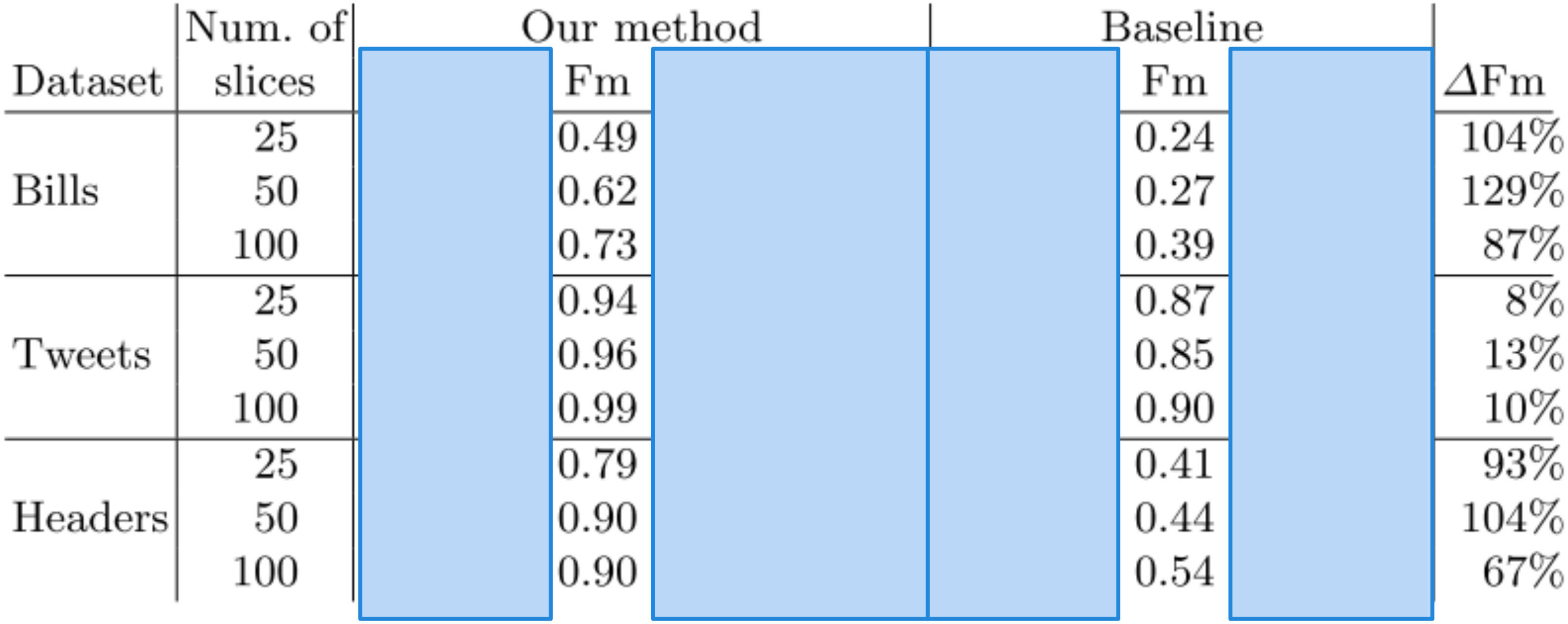


Computer

Automatic Synthesis of Regular Expressions from Examples

  - “Much better” than earlier regex learning proposals (for text extraction)

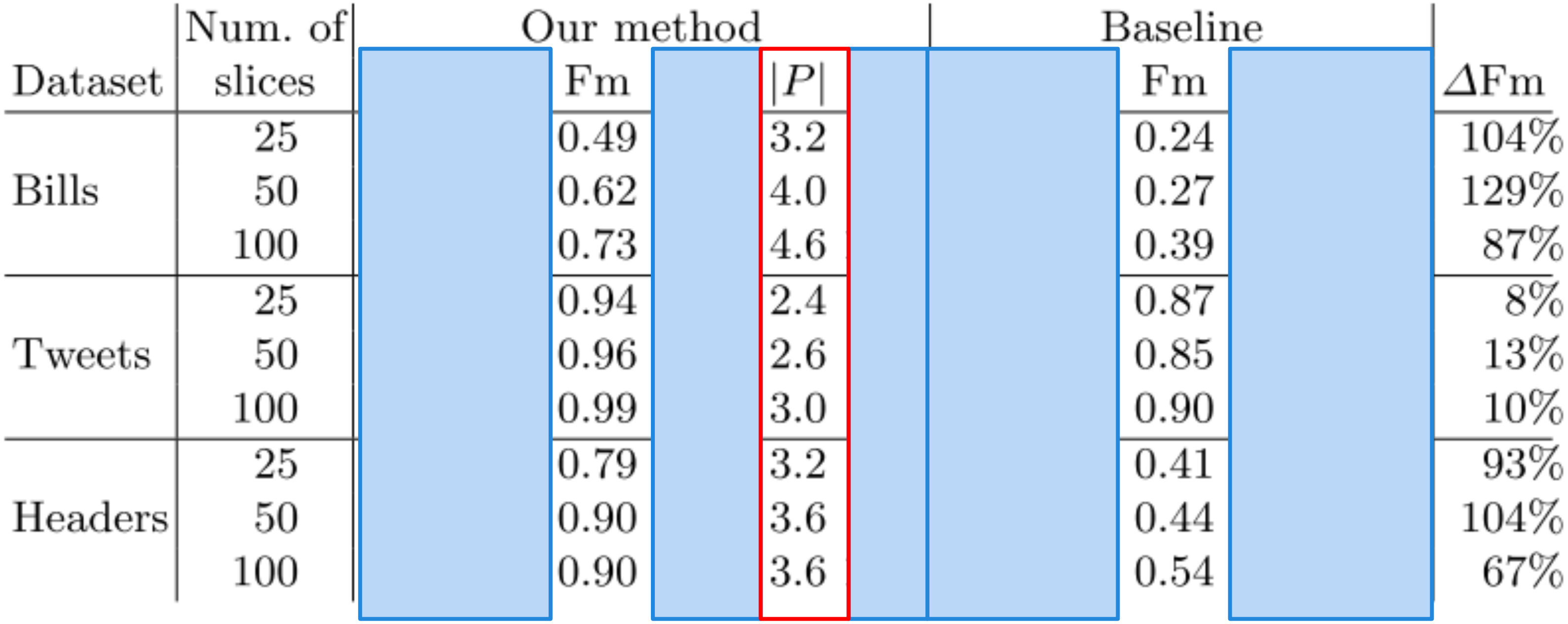
# Key results: F-measure



| Dataset | Num. of slices | Our method Fm | Baseline Fm | ΔFm |
|---|---|---|---|---|
| Bills | 25 | 0.49 | 0.24 | 104% |
| | 50 | 0.62 | 0.27 | 129% |
| | 100 | 0.73 | 0.39 | 87% |
| Tweets | 25 | 0.94 | 0.87 | 8% |
| | 50 | 0.96 | 0.85 | 13% |
| | 100 | 0.99 | 0.90 | 10% |
| Headers | 25 | 0.79 | 0.41 | 93% |
| | 50 | 0.90 | 0.44 | 104% |
| | 100 | 0.90 | 0.54 | 67% |

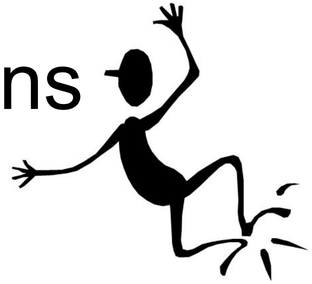
- F-measure
  - Significant improvement
  - Absolute values “practically useful”

# Key results: Multipattern

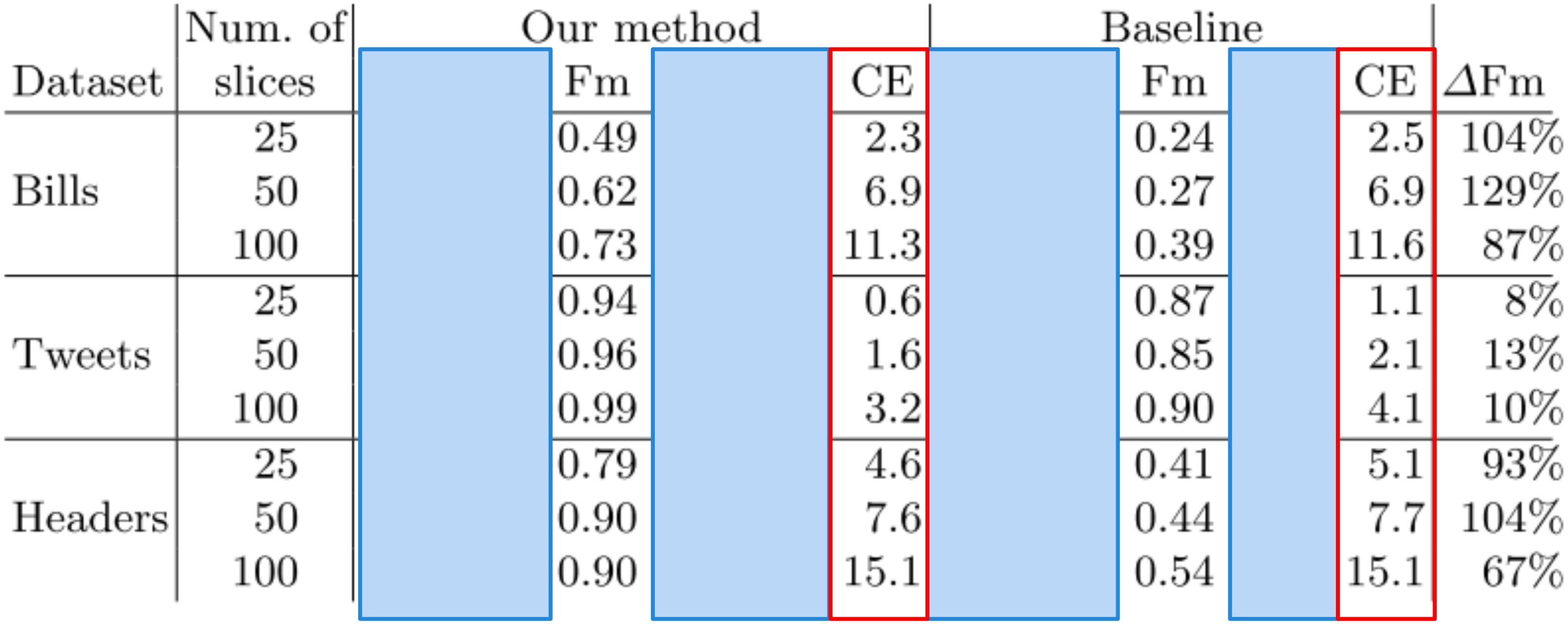


| Dataset | Num. of slices | Our method | | Baseline | |
|---|---|---|---|---|---|
| | | Fm | $\lvert P\rvert$ | Fm | ΔFm |
| Bills | 25 | 0.49 | 3.2 | 0.24 | 104% |
| | 50 | 0.62 | 4.0 | 0.27 | 129% |
| | 100 | 0.73 | 4.6 | 0.39 | 87% |
| Tweets | 25 | 0.94 | 2.4 | 0.87 | 8% |
| | 50 | 0.96 | 2.6 | 0.85 | 13% |
| | 100 | 0.99 | 3.0 | 0.90 | 10% |
| Headers | 25 | 0.79 | 3.2 | 0.41 | 93% |
| | 50 | 0.90 | 3.6 | 0.44 | 104% |
| | 100 | 0.90 | 3.6 | 0.54 | 67% |

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



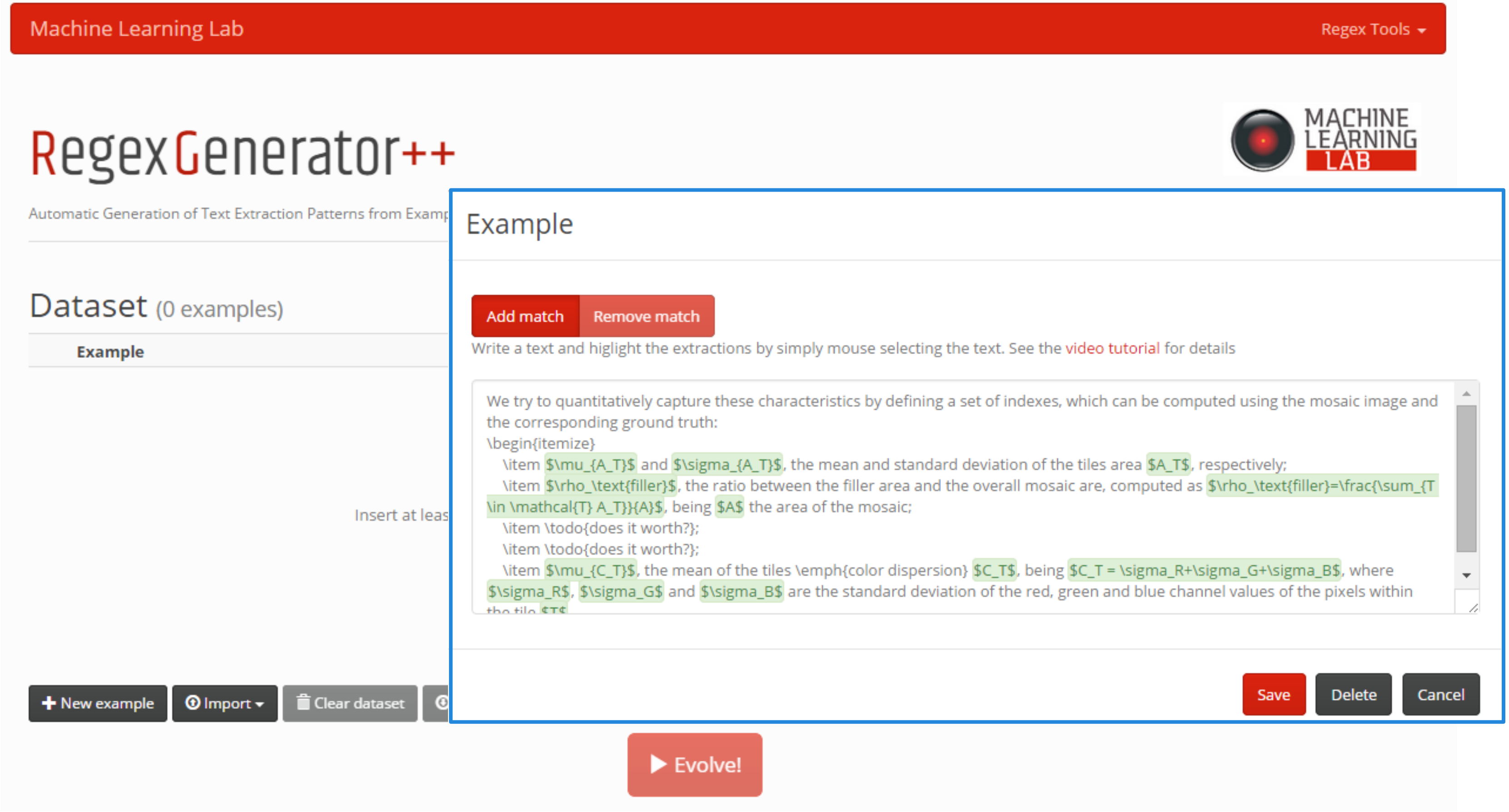
# Key results: Computational effort



| Dataset | Num. of slices | Our method | | Baseline | | |
|---|---|---|---|---|---|---|
| | | Fm | CE | Fm | CE | ΔFm |
| Bills | 25 | 0.49 | 2.3 | 0.24 | 2.5 | 104% |
| | 50 | 0.62 | 6.9 | 0.27 | 6.9 | 129% |
| | 100 | 0.73 | 11.3 | 0.39 | 11.6 | 87% |
| Tweets | 25 | 0.94 | 0.6 | 0.87 | 1.1 | 8% |
| | 50 | 0.96 | 1.6 | 0.85 | 2.1 | 13% |
| | 100 | 0.99 | 3.2 | 0.90 | 4.1 | 10% |
| Headers | 25 | 0.79 | 4.6 | 0.41 | 5.1 | 93% |
| | 50 | 0.90 | 7.6 | 0.44 | 7.7 | 104% |
| | 100 | 0.90 | 15.1 | 0.54 | 15.1 | 67% |

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

# http://regex.inginf.units.it



RegexGenerator++

Automatic Generation of Text Extraction Patterns from Examp

Dataset (0 examples)

Example

Insert at leas

Example

Write a text and higlight the extractions by simply mouse selecting the text. See the video tutorial for details

```
We try to quantitatively capture these characteristics by defining a set of indexes, which can be computed using the mosaic image and the corresponding ground truth:
\begin{itemize}
    \item $\mu_{A_T}$ and $\sigma_{A_T}$, the mean and standard deviation of the tiles area $A_T$, respectively;
    \item $\rho_\text{filler}$, the ratio between the filler area and the overall mosaic are, computed as $\rho_\text{filler}=\frac{\sum_{T \in \mathcal{T} A_T}{A}$, being $A$ the area of the mosaic;
    \item \todo{does it worth?};
    \item \todo{does it worth?};
    \item $\mu_{C_T}$, the mean of the tiles \emph{color dispersion} $C_T$, being $C_T = \sigma_R+\sigma_G+\sigma_B$, where $\sigma_R$, $\sigma_G$ and $\sigma_B$ are the standard deviation of the red, green and blue channel values of the pixels within the tile $T$
```

- Source code will be made public soon (GitHub)

# Thanks for your attention

MACHINE LEARNING LAB

University of Trieste, Italy

`http://machinelearning.inginf.units.it`

follow us on twitter @MaleLabTs