Seer: Automatically Learning Rules for Information Extraction

Maeda Hanafi (maeda.hanafi@nyu.edu), Azza Abouzied (azza@nyu.edu),
Laura Chiticariu (chiti@us.ibm.com), Yunyao Li (yunyaoli@us.ibm.com)

PROBLEM: Information Extraction

Extracting information from large collections of documents is time-consuming and complicated.

- Rule-based method: Users have to program their own extraction scripts with R, Python, AQL, etc. Users may only have minimal examples of what to extract.
- Machine learning method: A lot of data is needed for training, which isn't always possible.

SOLUTION: Seer

- Users provide examples by highlighting areas of the text they wish to extract.
- Seer suggests and refine extraction scripts based on example text from user. Seer learns rules and refinements to suggest to the user.
- Seer creates and executes extraction rules in IBM's extraction language, AQL.

1 HIGHLIGHT

The user highlights examples of what to extract.

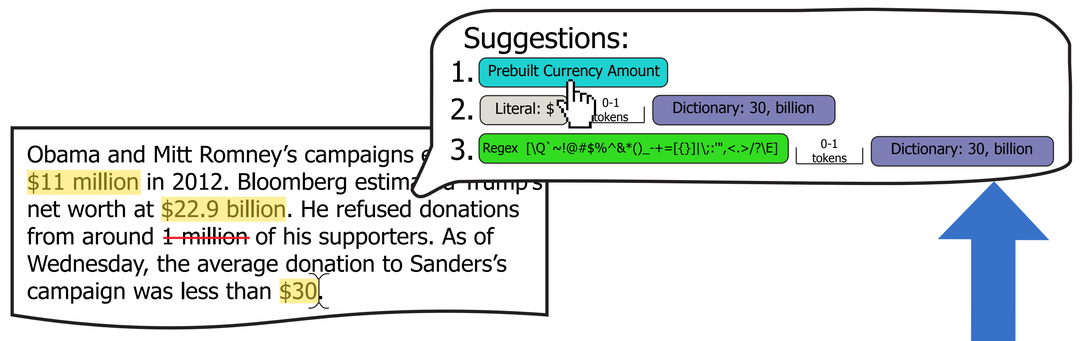
Obama and Mitt Romney's campaigns each raised $11 billion in 2012. Bloomberg estimated Trump's net worth at $22.9 billion. He refused donations from around ~~1 million~~ of his supporters. As of Wednesday, the average donation to Sanders's campaign was less than $30.

Refine

Highlight more examples or cross out highlights.

3 EXTRACT

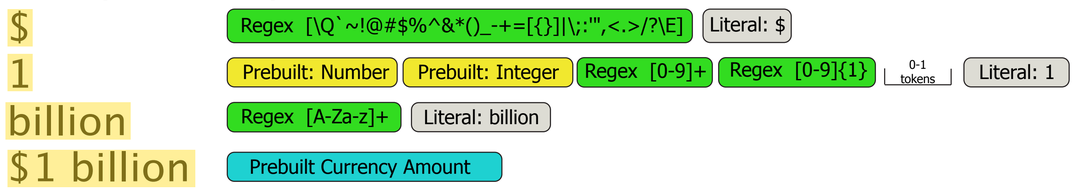
The rules are suggested. User selects and accepts a rule by clicking on it. The rule is run on all documents. The extraction results are highlighted.



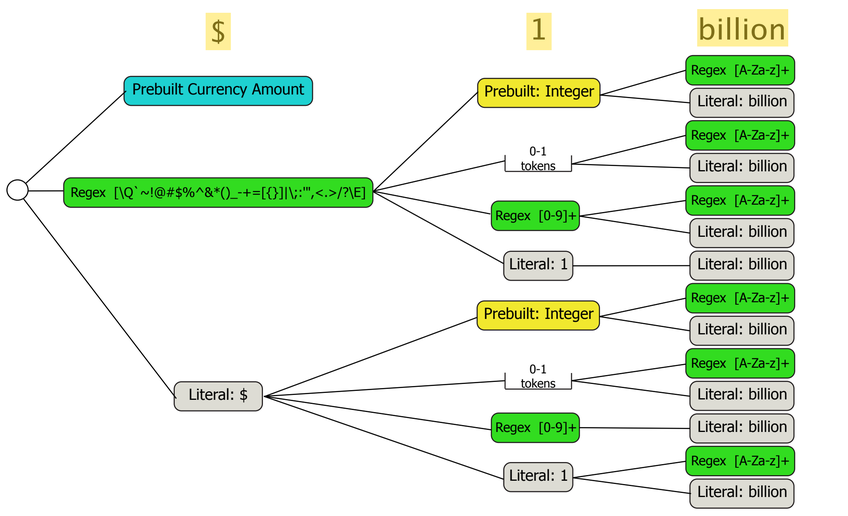
2 LEARN EXTRACTION RULES

a) The goal of learning is to show the user a list of distinct rules covering all examples.

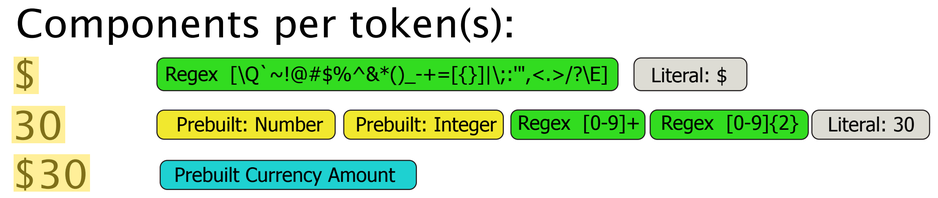
Components per token(s):

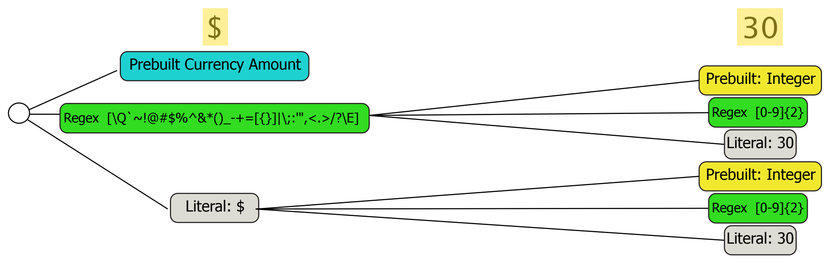


- Build rules from the components, and store it in a tree structure.



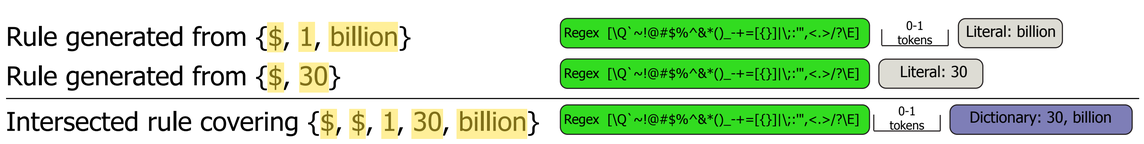
Components per token(s):

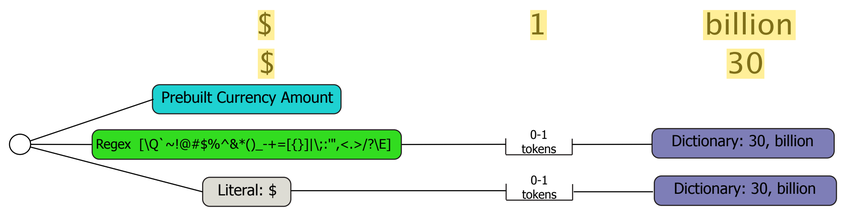




b) Intersect the trees to generate rules that capture all of the positive examples.

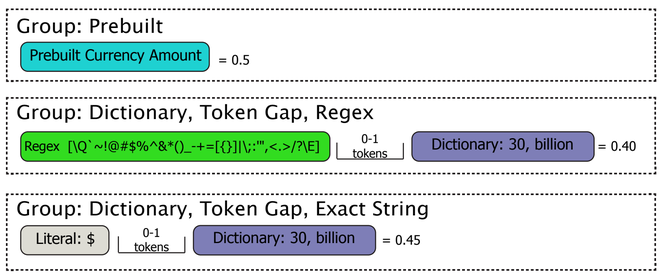
- Intersecting two trees means intersecting the rules of the trees.
- Two rules can be intersected if the order and types of the components are similar.
  Special case: Literals can be intersected into a dictionary component.





c) To ensure diversity in the final suggestions of rules, the similar rules are grouped together, and the rule with the highest score from each group are suggested to the user.

- Each group contains rules that have the same components.
- Grouped rules:



- The score of a rule depends on:

  - The inherent semantics of the component's originating token:
    Prebuilt: Integer is preferred over a 0-1 tokens to describe a number, 30

  - Occurence of a component's tokens:
    Since the token $ occurs more than once across all positive examples, Literal: $ is preferred over Regex [\Q`~!@#$%^&*()_-+={}]|\;:'",<.>/?\E]

  - Tokens from a negative example that can be captured by the rule's components:
    Regex [0-9]+ is preferred over Regex [0-9]{1} since it can capture 1, which is a token in the negative example ~~1 million~~

- Final rules to suggest to the user: