LaSEWeb: Automating Search Strategies over Semi-Structured Web Data

Olesandr Polozov
University of Washington
polozov@cs.washington.edu

Sumit Gulwani
Microsoft Research
sumitg@microsoft.com

Motivation
A significant percent of search queries constitute repetitive tasks. Two most common examples are:
1. Batch data extraction, done by end-users.
2. Development of micro-segments of factoid question answering in search engines.

Typical solutions involve:
- Using a structured database (e.g., FreeBase) (limited in content; hard-coded; time-insensitive)
- Writing a data mining script (fragile; inapplicable for end-users)

Both solutions do not preserve any of the following end-users’ search process patterns:
- Checking multiple webpages/answer candidates
- Exploring the context related to each answer
- Utilizing a semi-structured webpage format

LaSEWeb: Automating Search Strategies over Semi-Structured Web Data

LaSEWeb Query Language
A semantic scripting language for repetitive Web mining, based on the patterns in humans’ search strategies. The set of patterns is modular, extensible, and is implemented using the state-of-the-art ML/NLP algorithims.

LaSEWeb Search Algorithm

Problem definition
A Web program \( P \) is a parameterized query that:
- takes a tuple of user query arguments \( \vec{\nu} \)
- and returns a set of:
  - answer strings \( \alpha_i \)
  - ranked by their confidence \( \beta_i \)
  - with a set of the corresponding source URLs \( U_i \).

Workflow

LaSEWeb engine

Visual interpreter

Structural interpreter

Linguistic interpreter

LaSEWeb Query Language

LaSEWeb query
\[
\mathcal{L} = \text{FW}(\mathcal{B}, \Psi) \mid \mathcal{Q} \mid \mathcal{V} \cup \mathcal{Q}
\]

Visual expression
\[
\mathcal{B} = S \mid \text{Union}(B_1, B_2) \mid \eta : B
\]

Visual constraint
\[
\Psi = \text{Nearby}(\eta_1, \eta_2) \mid \text{Emphasized}(\eta) \mid \text{Layout}(\eta_1, \eta_2, d)
\]

Linguistic expression
\[
\mathcal{L} = \text{Ling}(\mathcal{E}, \Phi) \mid \text{Cl}_{1, 2}, \mathcal{C}_3 \mid \text{AttrLookup}(\mathcal{C}_1, \mathcal{C}_2)
\]

Linguistic constraint
\[
\Phi = \text{Sentence}(\epsilon_1, \epsilon_2) \mid \text{Regexp}(\epsilon_1) \mid \text{Word}(\epsilon_1)
\]

Visual patterns:
- webpage layout, colors, style, HTML, CSS
- Describe stylistic webpage properties, as seen by end-users
- Interpretation: rendering & DOM analysis

Structural patterns:
- implicit content schema, tables, lists
- Describe relational patterns on implicit tables
- Interpretation: table detection, plaintext analysis using PBE [1]

Linguistic patterns:
- text syntax, semantics, language, regexes
- Describe fuzzy semantic subexpressions of the webpage text
- Interpretation: POS tagging, sentence parsing, entity recognition [2-5], synonymy detection [6]

Example

LaSEWeb Query Language

LaSEWeb Search Algorithm

Given:
- Seed query function \( q(\vec{\nu}) \)
- Similarity metric \( \sigma(s_1, s_2) \)
- LaSEWeb query \( \mathcal{Q} \)
- Answer label \( \mathcal{L}_a \)

Do:
1. Search the Web for top-\( k \) relevant webpages using \( q(\vec{\nu}) \).
2. Match the LaSEWeb query on each webpage and extract \( \mathcal{L}_a \).
3. Cluster the resulting answer candidates based on similarity \( \sigma \).
4. Rank the clusters and select representative answers.

LaSEWeb Search Algorithm

Evaluation

Example

Micro-segments: 100,000+ user queries across 7 micro-segments from Bing search logs. Precision evaluated through random sampling, 95% in top-3 results. Average execution time: 5 sec/page.

Batch data extraction: 5 academic Web mining scenarios, precision and recall evaluated manually.