Wrapper Learning

Craig Knoblock
University of Southern California

This presentation is based on slides prepared by Ion Muslea

Wrapper Induction

Problem description:
- Web sources present data in human-readable format
  - take user query
  - apply it to data base
  - present results in "template" HTML page
- To integrate data from multiple sources, one must first extract relevant information from Web pages
- Task: learn extraction rules based on labeled examples
  - Hand-writing rules is tedious, error prone, and time consuming

Example of Extraction Task

NAME: Casablanca Restaurant
STREET: 220 Lincoln Boulevard
CITY: Venice
PHONE: (310) 392-5751

In this part of the lecture ... 

- Wrapper Induction Systems
  - WIEN:
    - The rules
    - Learning WIEN rules
  - SoftMealy
- The STALKER approach to wrapper induction
  - The rules
  - The ECTs
  - Learning the rules
- Wrapper validation and maintenance

WIEN [Kushmerick et al ‘97, ‘00]

- Assumes: items are always in fixed, known order
  ... Name: J. Doe; Address: 1 Main; Phone: 111-111. <p>
  ... Name: E. Poe; Address: 10 Pico; Phone: 777-1111. <p>
- Introduces several types of wrappers
  - LR:
    Name: ... Addr: ... Phone: ...
Wrapper Types

- LR
  - L and R delimit each of the k attributes
- HLRT
  - Two additional strings:
    - H marks the end of the header
    - T marks the beginning of the tail
- BELR
  - B & E mark the beginning and end of each tuple
- HBELRT
  - ??

Rule Learning

- Machine learning:
  - Goal: Find an instance of the given wrapper type that covers the given examples
- INPUT:
  - Labeled examples: training & testing data
  - Admissible rules (hypotheses space)
  - Search strategy
- Desired output:
  - Rule that performs well both on training and testing data
- Termination
  - Train on sufficient data to be provably approximately correct (PAC)

Learning LR extraction rules

- Admissible rules:
  - prefixes & suffixes of items of interest
- Search strategy:
  - start with shortest prefix & suffix, and expand until correct

Learning LR extraction rules

- Admissible rules:
  - prefixes & suffixes of items of interest
- Search strategy:
  - start with shortest prefix & suffix, and expand until correct
Learning LR extraction rules

• Admissible rules:
  • prefixes & suffixes of items of interest
• Search strategy:
  • start with shortest prefix & suffix, and expand until correct

Labeling Data

• Instead of labeling all of the data, use recognizers to find instances of a particular attribute
• Recognizers may be:
  • Perfect
    • Accept all positive instances and reject all negatives
  • Incomplete
    • Reject all negative instances but reject some positives
  • Unsound
    • Accept all positive, but accept some negatives
  • Unreliable
    • Reject some positive instances and accept some negatives
• Combine the constraints on the ordering of attributes with the information on the type of recognizer
  • E.g., if a perfect recognizer says that position 15-19 is the year and an unsound recognizer says that 18-19 is the age, then the later information would be considered a false positive

Summary

• Advantages:
  • Fast to learn & extract
  • Some sources could be labeled automatically given an appropriate set of recognizers
• Drawbacks:
  • Cannot handle permutations and missing items
  • Entire page must be labeled
  • Requires large number of examples

In this part of the lecture ...

• Wrapper Induction Systems
  • WIEN:
    • The rules
    • Learning WIEN rules
  • The STALKER approach to wrapper induction
    • The rules
    • The ECTs
    • Learning the rules
• Wrapper validation and maintenance

STALKER [Muslea et al, ’98 ’99 ’01]

• Hierarchical wrapper induction
  • Decomposes a hard problem in several easier ones
  • Extracts items independently of each other
  • Each rule is a finite automaton
**STALKER: The Wrapper Architecture**

STALKER: The Wrapper Architecture

1. **Information Extractor**
   - Query
   - Data

2. **EC Tree**

3. **Extraction Rules**

**Extraction Rules**

**Extraction rule**: sequence of *landmarks*

- \( \text{SkipTo(Phone)} \) \( \text{SkipTo(} <i> \text{)} \) \( \text{SkipTo(} </i> \text{)} \)

**Name: Joel’s**
- **Phone**: \( (310) 777-1111 \)
- **Review**: …

**More about Extraction Rules**

- **Name: Joel’s**
  - **Phone**: \( (310) 777-1111 \)

**Name: Kim’s**
- **Phone (toll free)**: \( (800) 757-1111 \)
- **Review**: …

- **Name: Joel’s**
  - **Phone**: \( (310) 777-1111 \)

**Start**: EITHER
- \( \text{SkipTo(Phone): } <i> \) \)
- OR \( \text{SkipTo(Phone): } </i> \)

**Learning the Extraction Rules**

**Example of Rule Induction**

**Training Examples**:

- **Name: Del Taco**
  - **Phone (toll free)**: \( (800) 123-4567 \)
- **Cuisine**: …
- **Name: Burger King**
  - **Phone**: \( (310) 987-9876 \)
  - **Cuisine**: …

**The Embedded Catalog Tree (ECT)**

- **Name: KFC**
- **Cuisine**: Fast Food

- **Locations**:
  - **Venice**: \( (310) 123-4567 \)
  - \( (800) 888-4412 \)
  - **City**: L.A.
  - \( (213) 987-6543 \)
  - **Encino**: \( (818) 999-4567 \)
  - \( (888) 727-3331 \)

**Inductive Learning System**

**EC Tree**
Example of Rule Induction

Training Examples:

Name: Del Taco <p> Phone (toll free): <b> (  800  ) 123-4567 </b> <p>Cuisine: ...

Name: Burger King <p> Phone : ( 310 ) 987-9876 <p> Cuisine: ...

Initial candidate:  

Example of Rule Induction

Training Examples:

Name: Del Taco <p> Phone (toll free): <b> (  800  ) 123-4567 </b> <p>Cuisine: ...

Name: Burger King <p> Phone : ( 310 ) 987-9876 <p> Cuisine: ...

Initial candidate:  

Active Learning & Information Agents

• Active Learning
  • Idea: system selects most informative exs. to label
  • Advantage: fewer examples to reach same accuracy

• Information Agents
  • One agent may use hundreds of extraction rules
  • Small reduction of examples per rule => big impact on user
  • Why stop at 95-99% accuracy?
  • Select most informative examples to get to 100% accuracy

Which example should be labeled next?

Training Examples

Name: Joel's <p> Phone: (310) 777-1111 <p>Review: The chef...

Name: Kim's <p> Phone: (213) 757-1111 <p>Review: Korean...

Multi-view Learning

Two ways to find start of the phone number:

SkipTo( Phone: )  

Unlabeled Examples

Name: Chez Jean <p> Phone: (310) 666-1111 <p> Review: ...

Name: Burger King <p> Phone: (818) 789-1211 <p> Review: ...

Name: Café del Rey <p> Phone: (310) 111-1111 <p> Review: ...

Name: KFC <p> Phone: (800) 111-7171 <p> Review: Fried chicken...

SkipTo( Phone: )  

BackTo( ( Number ) )

Name: KFC <p> Phone: (310) 111-1111 <p> Review: Fried chicken...
**Co-Testing**

- RULE 1
  - Labeled data
  - +
  - Unlabeled data

- RULE 2

**Co-Testing for Wrapper Induction**

<table>
<thead>
<tr>
<th>SkipTo(Phone:)</th>
<th>BackTo(Number:)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name: Joel’s</td>
<td>Phone: (310) 777-1111</td>
</tr>
<tr>
<td>Name: Kim’s</td>
<td>Phone: (213) 757-1111</td>
</tr>
<tr>
<td>Name: Chez Jean</td>
<td>Phone: (310) 666-1111</td>
</tr>
<tr>
<td>Name: Burger King</td>
<td>Phone: (818) 789-1211</td>
</tr>
<tr>
<td>Name: Café del Rey</td>
<td>Phone: (310) 111-1111</td>
</tr>
<tr>
<td>Name: KFC</td>
<td>Phone: (800) 111-7171</td>
</tr>
</tbody>
</table>

**Not all queries are equally informative**

- SkipTo(Phone:)
- BackTo(Number:)

---

**Weak Views**

- Learn "content description" for item to be extracted
  - Too general for extraction
    - (Nmb) Nmb - Nmb can’t tell a phone number from a fax number
  - Useful at discriminating among query candidates
- Learned field description
  - Starts with: (Nmb)
  - Ends with: Nmb - Nmb
  - Contains: Nmb Punct
  - Length: [6,6]

---

**Naïve & Aggressive Co-Testing**

- Naïve Co-Testing:
  - Query: randomly chosen contention point
  - Output: rule with fewest mistakes on queries

- Aggressive Co-Testing:
  - Query: contention point that most violates weak view
  - Output: committee vote (2 rules + weak view)

---

**Empirical Results: 33 Difficult Tasks**

- 33 most difficult of the 140 extraction tasks
  - Each view: > 7 labeled examples for best accuracy
  - At least 100 examples for task
Results in 33 Difficult Domains

Summary

- Advantages:
  - Powerful extraction language (e.g., embedded list)
  - One hard-to-extract item does not affect others
- Disadvantage:
  - Does not exploit item order (sometimes may help)

Discussion

- Basic problem is to learn how to extract the data from a page
- Range of techniques that vary in the
  - Learning approach
  - Rules that can be learned
  - Efficiency of the learning
  - Number of examples required to learn
- Regardless, all approaches
  - Require labeled examples
  - Are sensitive to changes to sources

In this part of the lecture ...

- Wrapper Induction Systems
  - WIEN:
    - The rules
    - Learning WIEN rules
- The STALKER approach to wrapper induction
  - The rules
  - The ECTs
  - Learning the rules
- Wrapper validation and maintenance
Wrapper Maintenance

Problem
- Landmark-based extraction rules are fast and efficient...but they rely on stable Web Page layout.
- If the page layout changes, the wrapper fails!
- Unfortunately, the average site on the Web changes layout more than twice a year.
- Requirement: Need to detect changes and automatically re-induce extraction rules when layout changes

Learning Regular Expressions [Goan, Benson, & Etzioni, 1996]
- Character level description of extracted data
- Based on ALERGIA [Carrasco and Oncina, 1994]
  - Stochastic grammar induction algorithm
  - Merges too many states resulting in over-general grammar
- WIL reduced faulty merges by imposing syntactic categories:
  - Number, lower upper, and delimit
  - Only merges when nodes contain the same syntactic category
  - Requires large number of examples to learn
  - Computationally expensive

Learning Global Properties for Wrapper Verification [Kushmerick, 1999]
- Each data field described by global numeric features
  - Word count, average word length, HTML density, alphabetic density
- Computationally efficient learning
- HTML density alone could account for almost all changes on test set
- Large number of false negatives on real changes to web sources [Lerman, Knoblock, Minton, 2002]

Learning Data Prototypes [Lerman & Minton, 2000]
- Approach to learning the structure of data
- Token level syntactic description
  - Descriptive but compact
  - Computationally efficient
- Structure is described by a sequence (pattern) of general and specific tokens.
- Data prototype = starting & ending patterns
  - STREET_ADDRESS
  - 220 Lincoln Blvd
  - 420 S Fairview Ave
  - 2040 Sawtelle Blvd
  - start with:
  - end with:
  - _NUM _CAPS Blvd
  - _CAPS _CAPS

Token Syntactic Hierarchy
- Tokens = words
- Syntactic types e.g., NUMBER, ALPHA
- Hierarchy of types allows generalization
- Extensible
  - new types
  - domain-specific information

Prototype Learning Algorithm
- No explicit negative examples
- Learn from positive examples of data
- Find patterns that
  - describe many of the positive examples of data
  - highly unlikely to describe a random token sequence (implicit negative examples)
  - are statistically significant patterns
- DataPro - efficient (greedy) algorithm
DataPro Algorithm

- Process examples
- Seed patterns
- Specialize patterns loop
  - Extend the pattern
    - find a more specific description
    - is the longer pattern significant given the shorter pattern?
  - Prune generalizations
    - is the pattern ending with general type significant given the patterns ending with specific tokens

Examples:
- 220 Lincoln Blvd
- 420 S Fairview Ave
- 2040 Sawtelle Blvd

Examples: PHONE

- starting patterns:
  - ( _NUM ) _NUM - _NUM
- ending patterns:
  - ( _NUM ) _NUM - _NUM

Examples:
- (310) 577 - 8182
- (310) 652 - 9770
- (310) 396 - 1179
- (310) 477 - 7242
- (626) 782 - 9779
- (310) 823 - 4446
- (323) 870 - 2872
- (310) 655 - 9380
- (310) 578 - 2293
- (310) 392 - 5751
- (805) 683 - 8864
- (310) 301 - 1004
- (626) 783 - 8123
- (310) 822 - 1513

Example: STREET_ADDRESS

- 13455 Maxella Ave
- 903 N La Cienega Blvd
- 110 Navy St
- 2040 Sawtelle Blvd
- 87 E Colorado Blvd
- 4325 Glencoe Ave
- 2525 S Robertson Blvd
- 998 S Robertson Blvd
- 523 Washington Blvd
- 220 Lincoln Blvd
- 420 S Fairview Ave
- 13490 Maxella Ave
- 363 S Fair Oaks Ave
- 4676 Admiralty Way

- starting patterns:
  - _NUM S _CAPS Blvd
  - _NUM _CAPS Ave
  - _NUM _CAPS
- ending patterns:
  - _NUM _CAPS _CAPS
  - _NUM S _CAPS Blvd
  - _NUM _CAPS Ave
  - _NUM _CAPS Blvd

Wrapper Verification

Data prototypes can be used for web wrapper maintenance applications.

- Automatically detect when the wrapper is no longer correctly extracting data from an information source
  - (Kushmerick 1999)

Results

- Monitored 27 wrappers (23 distinct sources)
- There were 37 changes over ~ 1 year
- Algorithm discovered 35/37 changes with 15 mistakes
  - 13 false positives
- Overall:
  - Average precision = 73%
  - Average recall = 95%
  - Average accuracy = 97%
Wrapper Reinduction

- Rebuild the wrapper automatically if it is not extracting data correctly from new pages
- Data extraction step
  Identify correct examples of data on new pages
- Wrapper induction step
  Feed the examples, along with the new pages, to the wrapper induction algorithm to learn new extraction rules

The Lifecycle of A Wrapper

Example Source Change

Whitepages Wrapper

Wrapper Applied to Changed Source

After Reinduction
### Results

- Monitored 10 distinct sources
- There were 8 changes over ~ 1 year
- Extracting examples:
  - 277/338 correct (82%) 
  - 31 false positives/30 false negatives
- Reinduction:
  - Average recall = 90%
  - Average precision = 80%

### Discussion

- Flexible data representation scheme
- Algorithm to learn description of data fields
- Used in wrapper maintenance applications

Limitations:

- Needs to be extended to lists and tables
- Excellent recall, but lower recall will precision in many false positives