Extracting and Querying Probabilistic Information From Text in BayesStore-IE

Daisy Zhe Wang, Michael J. Franklin, Minos Garofalakis\textsuperscript{2}, Joseph M. Hellerstein

University of California, Berkeley

Technical University of Crete\textsuperscript{2}
Information Extraction (IE)

- “We are pleased that today's agreement guarantees our corporation will maintain a significant and long term presence in the Big Apple,” McGraw-Hill president Harold McGraw III said in a statement.

---  From New York Times April 24, 1997
“We are pleased that today's agreement guarantees our corporation will maintain a significant and long term presence in the Big Apple," McGraw-Hill president Harold McGraw III said in a statement.

--- From New York Times April 24, 1997

Labels:
- Person
- Company
- Location
- Other
“We are pleased that today's agreement guarantees our corporation will maintain a significant and long term presence in the Big Apple,” McGraw-Hill president Harold McGraw III said in a statement.

--- From New York Times April 24, 1997

Labels:
Person  Company  Location  Other
Standard IE Systems

Tokenization, Feature Extraction, Inference …

SELECT * 
FROM Entities

Extracted Entities

Statistical ML Packages

Relational DBMS
Problems

- Poor Performance
  - Exhaustive Batch Analysis
  - No Indexing, Parallelization, Caching, Optimization in Data Analysis in SML
- Information Loss
  - DB only store Top-1 Highest Probability Results
  - Prematurely discard Uncertainties and Probabilities
Probabilistic Database & Declarative IE

Probabilistic Database

Declarative IE
BayesStore-IE is a subsystem of BayesStore [VLDB08] for Probabilistic and Declarative IE.
BayesStore-IE is built on top of Postgres 8.4.
Main Contributions

- Efficient In-Database Graphical Models
  - Represent Models as First-class Objects
  - Implement Inference in Database
- Scale-up IE Queries
  - Query-Driven Extraction
  - Co-optimization
- Algorithms for Probabilistic IE Queries
  - Compute Top-k Results
  - Compute Result Distribution

Orders-of-Magnitude Speedup!

Reduce False Negatives by 80%!
Outline

• Introduction
• In-database Inference Algorithms [ICDE10, SIGMOD11]
• Scale-up declarative IE Queries [VLDB10]
• Hybrid Inference [SIGMOD11]
• Conclusion and Future Work: MADLib
A Graphical Model –
Conditional Random Fields (CRF)

Text (address string):
E.g., “2181 Shattuck North Berkeley CA USA”

CRF Model:

Possible Extraction Worlds:

<table>
<thead>
<tr>
<th></th>
<th>x</th>
<th>2181</th>
<th>Shattuck</th>
<th>North</th>
<th>Berkeley</th>
<th>CA</th>
<th>USA</th>
</tr>
</thead>
<tbody>
<tr>
<td>y1</td>
<td>apt. num</td>
<td>street name</td>
<td>city</td>
<td>city</td>
<td>state</td>
<td>country</td>
<td>(0.6)</td>
</tr>
<tr>
<td>y2</td>
<td>apt. num</td>
<td>street name</td>
<td>street name</td>
<td>city</td>
<td>state</td>
<td>country</td>
<td>(0.1)</td>
</tr>
</tbody>
</table>
BayesStore-IE Data Model

2181 Shattuck North
Berkeley CA USA

<table>
<thead>
<tr>
<th>docID</th>
<th>pos</th>
<th>token</th>
<th>LabelP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>2181</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Shattu</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>North</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>Berkeley</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>CA</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>USA</td>
<td></td>
</tr>
</tbody>
</table>

TokenTableP

<table>
<thead>
<tr>
<th>token</th>
<th>prevLabel</th>
<th>label</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shattuck</td>
<td>street num</td>
<td>street name</td>
<td>22</td>
</tr>
<tr>
<td>Shattuck</td>
<td>street num</td>
<td>street num</td>
<td>5</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Berkeley</td>
<td>street name</td>
<td>street name</td>
<td>10</td>
</tr>
<tr>
<td>Berkeley</td>
<td>street name</td>
<td>city</td>
<td>25</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

FactorTable
Relational and Inference Queries

- Relational Operators
  - Select, Project, Join
  - Aggregation

- Inference Operators
  - Top-k Inference
  - Marginal Inference

<table>
<thead>
<tr>
<th>docID</th>
<th>pos</th>
<th>token</th>
<th>Label^P</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>2181</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Shattuck</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>North</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>Berkeley</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>CA</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>USA</td>
<td></td>
</tr>
</tbody>
</table>

TokenTable^P

```
SELECT pos, token, top-k(Label^P)
FROM TokenTable^P
WHERE docID <= 10
```
Viterbi Implemented in SQL

Viterbi Dynamic Programming Algorithm:

\[
V(i, y) = \begin{cases} 
\max_y (V(i-1, y') + \sum_{k=1}^{K} \lambda_k \cdot f_k \cdot f(y, y', x_i)), & \text{if } i \geq 0 \\
0, & \text{if } i = -1.
\end{cases}
\]

```
<table>
<thead>
<tr>
<th>pos</th>
<th>street num</th>
<th>street name</th>
<th>city</th>
<th>state</th>
<th>country</th>
</tr>
</thead>
<tbody>
<tr>
<td>2181</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Shattuck</td>
<td>1</td>
<td>2</td>
<td>15</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>North</td>
<td>2</td>
<td>12</td>
<td>24</td>
<td>21</td>
<td>18</td>
</tr>
<tr>
<td>Berkeley</td>
<td>3</td>
<td>21</td>
<td>32</td>
<td>32</td>
<td>30</td>
</tr>
<tr>
<td>CA</td>
<td>4</td>
<td>29</td>
<td>40</td>
<td>38</td>
<td>42</td>
</tr>
<tr>
<td>USA</td>
<td>5</td>
<td>39</td>
<td>47</td>
<td>46</td>
<td>46</td>
</tr>
</tbody>
</table>
```
Viterbi Implemented in SQL [ICDE10]

Viterbi Dynamic Programming Algorithm:

\[
V(i, y) = \begin{cases} 
\max_{y'}(V(i-1, y')) + \sum_{k=1}^{K} \lambda_k \cdot f_k \cdot f(y, y', x_i), & \text{if } i \geq 0 \\
0, & \text{if } i = -1. 
\end{cases}
\]
Example Queries

<table>
<thead>
<tr>
<th>SELECT</th>
<th>( Top-k(D1.docID, D2.docID) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>FROM</td>
<td>( Emails^P \ D1, Emails^P \ D2 )</td>
</tr>
<tr>
<td>WHERE</td>
<td>( D1.docID \neq D2.docID ) and ( D1.Label^P = D2.Label^P = \text{‘person’} ) and ( D1.token = D2.token )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SELECT</th>
<th>( Marginal(D1.docID, D2.docID, \text{exist}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>FROM</td>
<td>( Emails^P \ D1, Emails^P \ D2 )</td>
</tr>
<tr>
<td>WHERE</td>
<td>( D1.docID \neq D2.docID ) and ( D1.Label^P = D2.Label^P = \text{‘person’} ) and ( D1.token = D2.token )</td>
</tr>
</tbody>
</table>
Why Different Inference Algorithms

Machine Learning

• Types of Inference
  • Marginal
  • Top-k

• Model Structures
  • Linear-chain
  • Tree-shaped
  • Cyclic
All Inference Algorithms Implemented

<table>
<thead>
<tr>
<th>Inference Algorithms</th>
<th>Top-k</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Linear</td>
<td>Tree</td>
<td>Cyclic</td>
<td>Linear</td>
<td>Tree</td>
<td>Cyclic</td>
<td></td>
</tr>
<tr>
<td>Viterbi (Max-Product)</td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sum-Product</td>
<td></td>
<td></td>
<td></td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCMC Gibbs</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>MCMC-MH</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>
In-Database MCMC Implementation [SIGMOD11]

- Iterative Implementation
- Set-oriented implementation
  - Window Functions
  - Array Data Types
  - UDF functions

- Guidelines for In-database implementation of statistical methods
Outline

- Introduction
- In-database Inference Algorithms [ICDE10, SIGMOD11]
- Scale-up declarative IE Queries [VLDB10]
- Hybrid Inference [SIGMOD11]
- Conclusion and Future Work: MADLib
Scale-up Probabilistic Queries:

Exhaustive Extraction
→ Query-Driven Extraction
Scale-up Probabilistic Queries: Exhaustive $\rightarrow$ Query-Driven Extraction

Techniques:

- Inverted Index $\rightarrow$ prune documents
- Minimize Model $\rightarrow$ prune model
- Early-stopping Viterbi $\rightarrow$ reduce inference computation

```
SELECT D2.token
FROM NYTimes$^P$ D1, NYTimes$^P$ D2
WHERE D1.docID = D2.docID and
      D1.token = 'Apple'
      and D1.Label$^P$ = 'company'
      and D2.Label$^P$ = 'company'
```
Inverted Index

```
SELECT D2.token
FROM NYTimes D1, NYTimes D2
WHERE D1.docID = D2.docID and
      D1.token = 'Apple' and D1.Label = 'company'
      and D2.Label = 'company'
```

$D_0 = \text{"it is what it is"}$, $D_1 = \text{"what is it"}$, $D_2 = \text{"it is a banana"}$

**Inverted Index Data Structure:**

- "a": {D2}
- "banana": {D2}
- "is": {D0, D1, D2}
- "it": {D0, D1, D2}
- "what": {D0, D1}
Minimizing Model

query → A → B → C

D → E

Sentence 1

X = tokens

Y = labels

Sentence 2

X = tokens

Y = labels

query
Minimizing Model

X = tokens
Y = labels

Sentence 2

query
Viterbi Early-Stopping Algorithm

SELECT `D2.token` FROM `NYTimes^P D1, NYTimes^P D2`
WHERE `D1.docID = D2.docID and D1.token = 'Apple'` and `D1.Label^P = 'company'` and `D2.Label^P = 'company'`
Evaluation 1: [Runtime] Scale-up Techniques

NYTimes Dataset: size 7GB, 1 million articles
Without Inverted Index runtime: ~12.8 hours (k=1)
Evaluation 2: [Runtime]
Scale-up Techniques

NYTimes Dataset: size 7GB, 1 million articles
Without Inverted Index runtime: ~12.8 hours (k=1)
Outline

- Introduction
- In-database Inference Algorithms [ICDE10, SIGMOD11]
- Scale-up declarative IE Queries [VLDB10]
- Hybrid Inference [SIGMOD11]
- Conclusion and Future Work: MADLib
Top-k with Skip-Chain Model

IBM Corp. said that IBM for IBM.

IBM Corp. said that HP for Apple.
Top-k with Skip-Chain CRF

General, but Approximate and Slow

- Max-Product/Viterbi
  - zero skip-edge (linear) at least one skip-edge (cyclic)
    - model instantiation
      - TokenTbl
      - Skip-Chain CRF
- Gibbs/MCMC-MH
  - model instantiation
    - TokenTbl
    - Skip-Chain CRF

(a) General, but Approximate and Slow
(b) Model instantiation
Marginal over Probabilistic Join

CEO  Bill  Gates  talked  about

E1  exist

assigned  by  Bill  Clinton  today
Marginal over Probabilistic Join

......

Bill Gates met with Bill

E2
exist

E1

......

assigned by Bill Clinton today

......
Marginal over Probabilistic Join

- only 1 cross-edge (tree)
- more than 2 cross-edges (cyclic)

model instantiation

- Join(token1=token2)
- Join(token1=token2 & label1=label2)

TokenTbl1  TokenTbl2  CRF1  CRF2

Sum-product

Union

MCMC Gibbs

General, but Approximate and Slow
## Preliminary Results

<table>
<thead>
<tr>
<th>Data Corpora</th>
<th>Skip-Chain CRF</th>
<th>Probabilistic Join</th>
</tr>
</thead>
<tbody>
<tr>
<td>NYTimes</td>
<td>5.0x</td>
<td>4.5x</td>
</tr>
<tr>
<td>Twitter</td>
<td>5.0x</td>
<td>2.6x</td>
</tr>
<tr>
<td>DBLP</td>
<td>1.0x</td>
<td>1.0x</td>
</tr>
</tbody>
</table>
Conclusion

- Efficient In-database Implementation of Inference Algorithms

- Scale-up IE Queries using Query-Driven Extraction achieves orders-of-magnitude Speedup

- Hybrid Inference Optimization can achieve up to 5x speed up on Twitter, NYTimes datasets
Related Work

MADLib

- started as collaboration between UC Berkeley and EMC/Greenplum
- MAD (Magnetic, Agile, Deep) skills [VLDB2009]
- open-source library (v0.2beta)
- mathematical, statistical, and machine learning methods for structured and unstructured data
- data-parallel implementations
- platforms: PostgreSQL and Greenplum
Thank you! ... Questions?

Daisy Zhe Wang Homepage: http://www.cs.berkeley.edu/~daisyw

BayesStore Project Page: http://www.cs.berkeley.edu/~daisyw/BayesStore.html

MAD Library (open source): http://madlib.net
Current and Future Work

- Information Extraction
  - Inference (Viterbi, MCMC)
  - Feature Extraction
  - Learning Algorithms (L-BFGS, online gradient decent)

- Reference Reconciliation
  - Query-Driven
  - Efficient and Scalable

- Cost-based Optimizer (Accuracy/Runtime tradeoffs)