

Declarative Information Extraction in a Probabilistic Database System

Daisy Zhe Wang

UC Berkeley

Information Extraction (IE)

- IE: extract snippets of structured information in text documents (e.g. name, company, address)

For years, [Microsoft Corporation](#) [CEO](#) [Bill Gates](#) was against open source. But today he appears to have changed his mind. "We can be open source. We love the concept of shared source," said [Bill Veghte](#), a [Microsoft VP](#). "That's a super-important shift for us in terms of code access."

[Richard Stallman](#), [founder](#) of the [Free Software Foundation](#), countered saying...

Select Name
From PEOPLE
Where Organization = 'Microsoft'

PEOPLE

<u>Name</u>	<u>Title</u>	<u>Organization</u>
Bill Gates	CEO	Microsoft
Bill Veghte	VP	Microsoft
Richard Stallman	Founder	Free Soft..

[Bill Gates](#)
[Bill Veghte](#) (from Cohen's IE tutorial, 2003)

IE in Database research

- Two Research Fronts
 - Declarative languages and Systems for managing IE tasks (e.g. DBLife, IBM SystemT, Yahoo PSOX,..)
 - Probabilistic Database to manage uncertain output of IE (e.g. MystiQ, Trio, MauveDB, MayBMS, BayesStore,...)
- Goal: Merge two directions
 - create a probabilistic database system that can manage IE tasks declaratively

Previous Attempts Storing IE Models in Probabilistic DB

- Gupta and Sarawagi (VLDB06) :
 - Statistical (CRF) Model to perform extraction
 - Extraction results store in ProbDB approximately
 - Optimize for accuracy and space trade-off

Figure 3: Four segmentations of the address string '52-A Goregaon West Mumbai 400 076' along with their probabilities.

Id	House_no	Area	City	Pincode	Prob
1	52	Goregaon West	Mumbai	400 062	0.1
1	52-A	Goregaon	West Mumbai	400 062	0.2
1	52-A	Goregaon West	Mumbai	400 062	0.5
1	52	Goregaon	West Mumbai	400 062	0.2

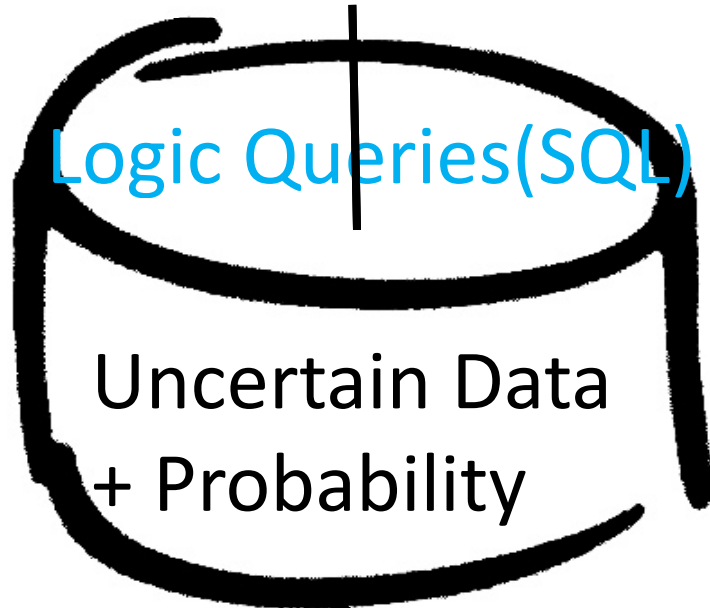
Figure 4: Segmentation-per-row model for the example in Figure 3.

Id	House_no	Area	City	Pincode
1	52 (0.3) 52-A (0.7)	Goregaon West (0.6) Goregaon (0.4)	Mumbai (0.6) West Mumbai (0.4)	400 062 (1.0)

Figure 5: One-row model for the example in Figure 3.

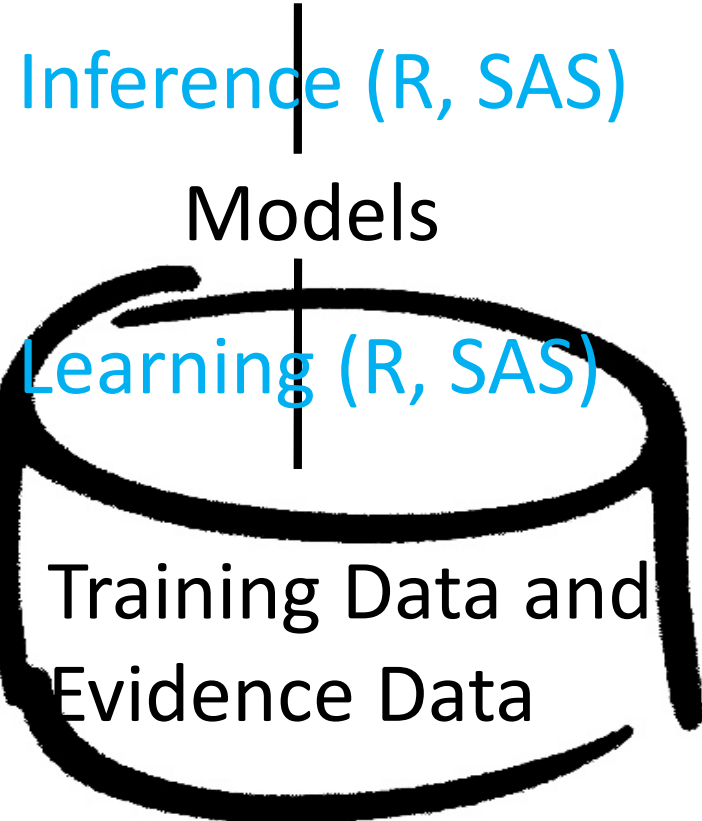
ProbDB and Statistical Data Analytics

Results +
Probability



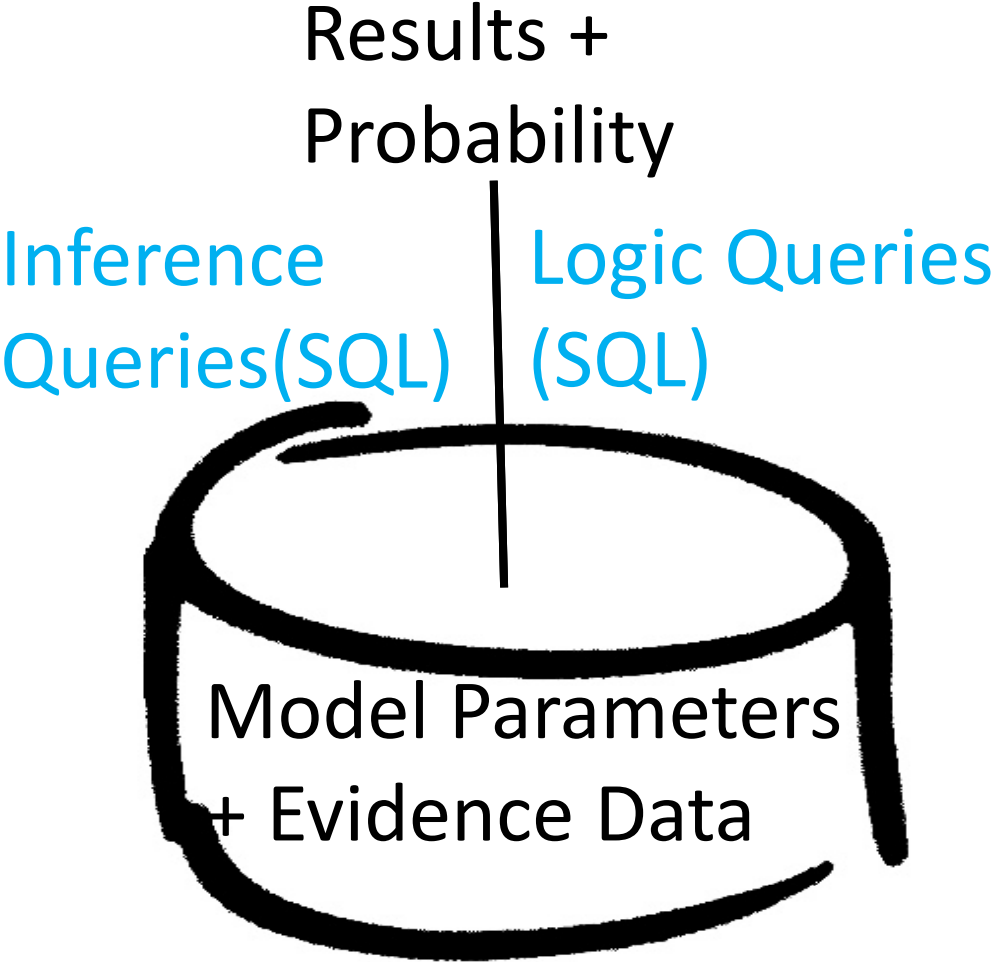
Probabilistic Database

Probability +
Uncertain Data



Statistical Data Analytics

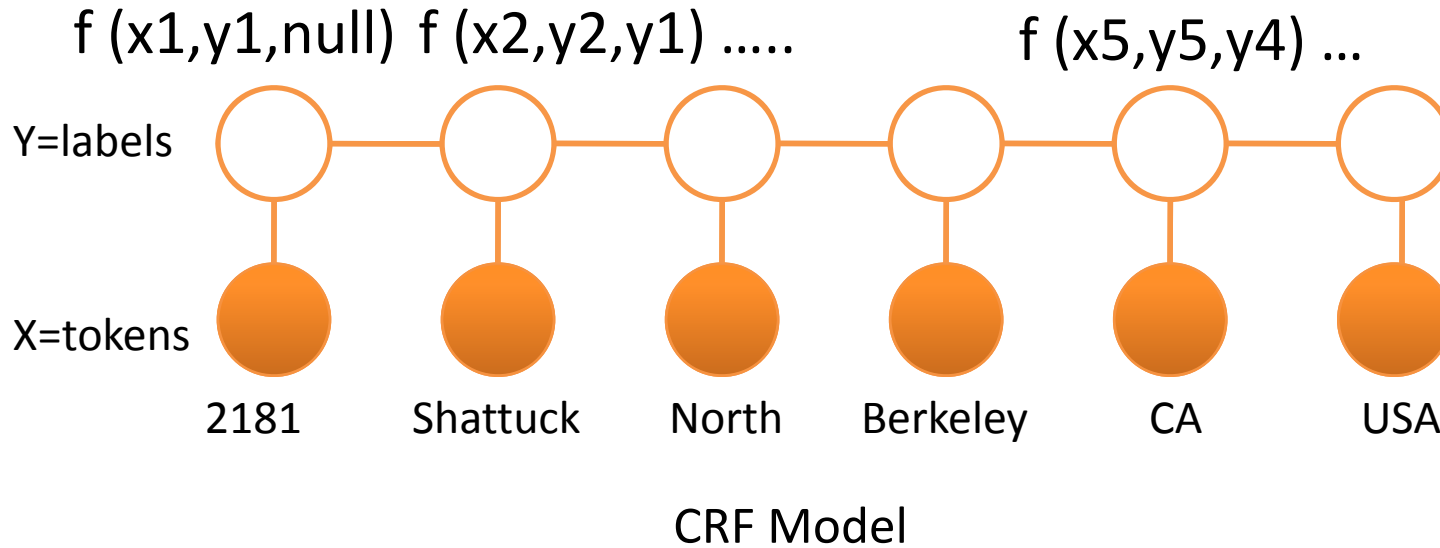
Our Attempt to Support CRF Models in ProbDB: the BayesStore Approach



Techniques

- Relational Representation of Inputs
 - CRF Model : factor tables
 - Text data : Inverted file tables
- Declarative Viterbi Inference in SQL
 - Viterbi as recursive query in SQL
 - ViterbiArray
- Query Optimizations
 - Selection and Inference
 - Join and Inference

Conditional Random Fields (CRF)

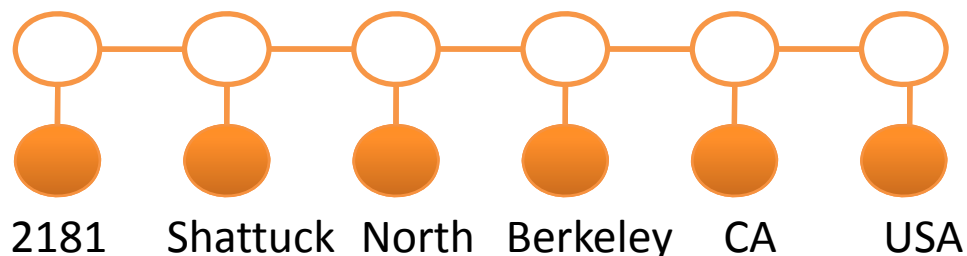


	apt. num	street num	street name	city	state	country	prob
y1	null	2181	Shattuck North	Berkeley	CA	USA	0.6
y2	2181	null	Shattuck	North Berkeley	CA	USA	0.1

Possible Worlds

CRF and Text Data in Relational Tables

- Text Data: Inverted file, one token per row
- CRF: factor table, one $\langle \text{token}, \text{prevLabel}, \text{label} \rangle$ triple per row



docID	pos	token	Label
1	0	2181	
1	1	Shattuck	
1	2	North	
1	3	Berkeley	
1	4	CA	
1	5	USA	

Token Table

token	prevLabel	label	score
DIGIT	null	street num	22
DIGIT	null	street name	5
...	
Berkeley	street	street name	10
Berkeley	street	city	25
..	

Factor Table

Top-k Inference on CRF

$$V(i, y) = \begin{cases} \max_{y'} (V(i-1, y') \\ + \sum_{k=1}^K \lambda_k f_k(y, y', x_i)), & \text{if } i \geq 0 \\ 0, & \text{if } i = -1. \end{cases} \quad (3)$$

pos	street num	street name	city	state	country
0	5	1	0	1	1
1	2	15	7	8	7
2	12	24	21	18	17
3	21	32	32	30	26
4	29	40	38	42	35
5	39	47	46	46	50

Viterbi Implemented in SQL

- Viterbi
 - Implemented as a UDF
 - WITH RECURSIVE : compute $V(i,y)$ from $V(i-1,y)$
 - Problem: 5 times slower than hand-tuned CRF open source library
- ViterbiArray
 - Represent factors $f(x,y,y')$ as an array
 - UDFs over array types for aggregation
 - Improve the main memory performance, Join efficiency, representation compactness
 - Results: Equal or better performance compare to open source CRF library

Results for Viterbi algorithms in SQL

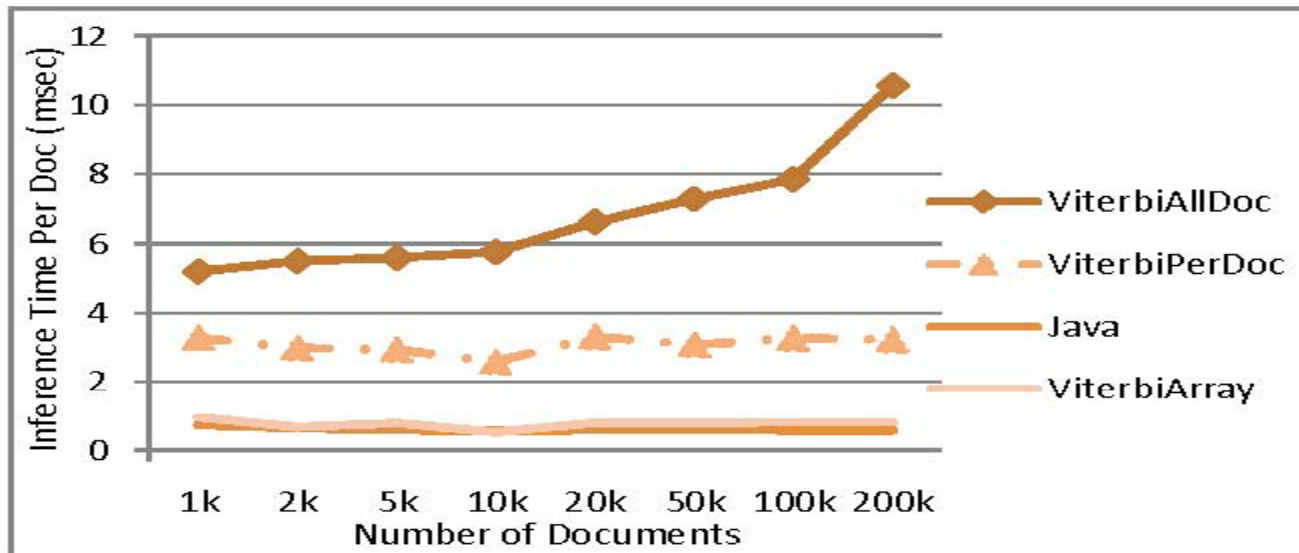


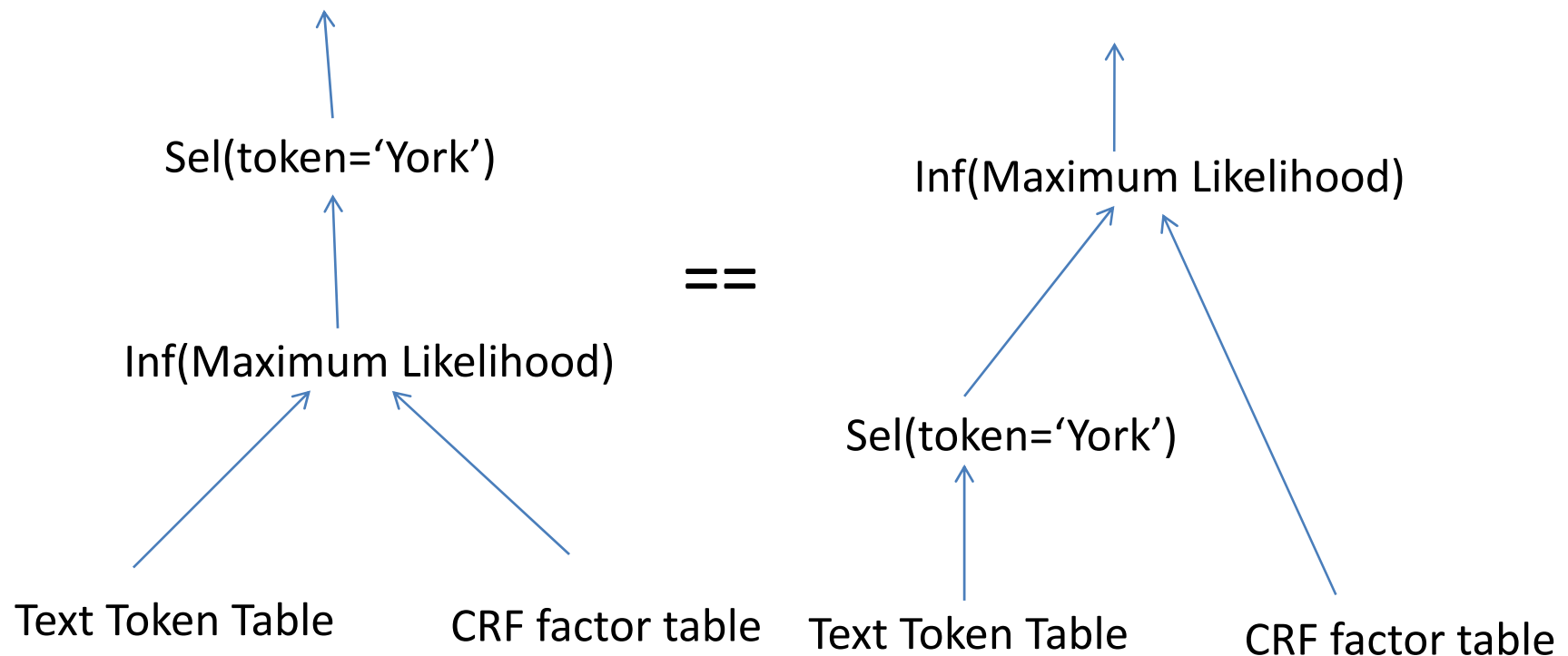
Figure 6: Average inference time (msec) for a single document for different implementations of the Viterbi algorithm.

dataset	ViterbiAllDoc	ViterbiPerDoc	ViterbiArray	Java
address	10.5 msec	3.2 msec	0.8 msec	0.5 msec
bib	1760.1 msec	175.1 msec	6.2 msec	16.2 msec

Figure 7: Average inference time per document (msec) for different Viterbi implementations on address and bib dataset.

Selection + Inference Optimization (1)

- Condition on token
 - Select * From (Select docID,pos,token,top-1(label) From Tokentbl) where token='York'



Selection + Inference Optimization (2)

- Condition on <pos,label>
 - Selection-aware Viterbi
 - Technique: early-stop dynamic programming
 - Select docID

From (Select docID,pos.,token,top-1(label)

From TokenTbl)

Where label='street name' and pos=0

pos	street num	street name	city	state	country
0	5	1	0	1	1
1	2	15	7	8	7
2	12	24	21	18	17
3	21	32	32	30	26
4	29	40	38	42	35
5	39	47	46	46	50

STOP!

Results for Selection-Inference Optimization

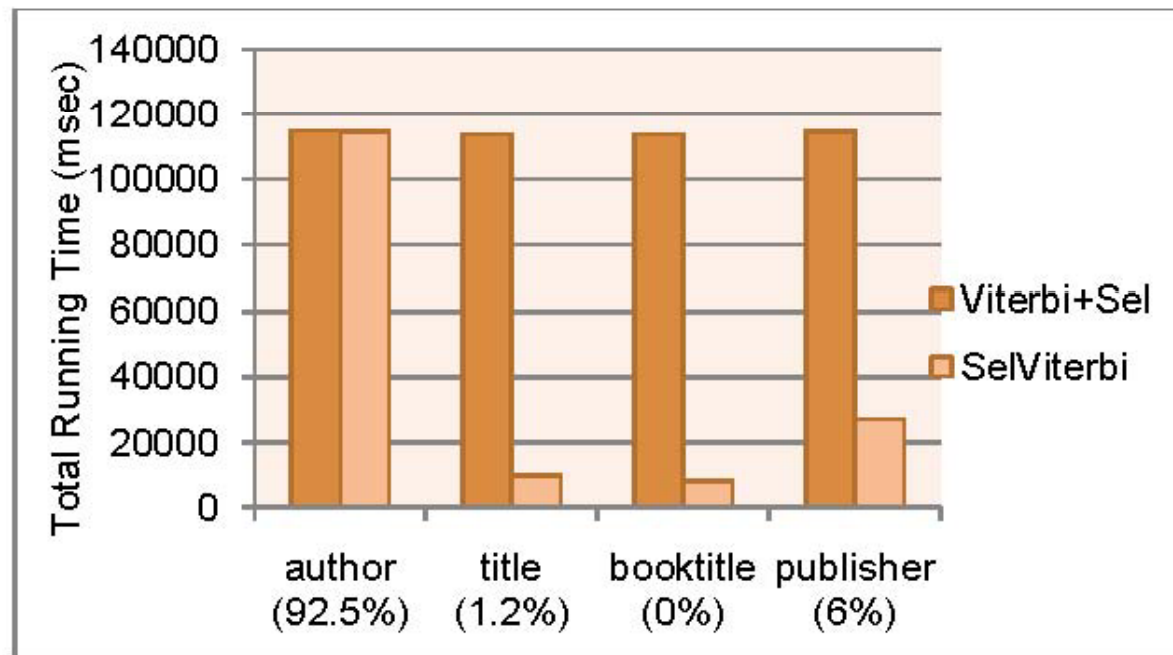


Figure 11: Total top-1 inference time (msec) for 18,000 bibliography entries with selection condition on label (x-axis) at position 0.

Join + Inference Optimization

$$\sigma_{\langle label1 \rangle}(\text{top1}(\text{TOKEN TBL}, \text{CRF1})) \bowtie_{\leq n} \sigma_{\langle label2 \rangle}(\text{top1}(\text{TOKEN TBL}, \text{CRF2})) \quad (\mathbf{V})$$

Join condition: follow-by n tokens

(followed within 5 tokens)

Plan A

Person Names

Telephone numbers

Identify telephone numbers
Within 5 tokens

Identify person names
Within 5 tokens

Plan B

Plan C

Extract Tokens to the right

Extract Tokens to the left

Person Names

Telephone numbers

Future Directions: Tie in Data Analytics into Probabilistic Databases

- Extensible API to plug in new statistical models for data analytics
- Queries over statistical models, PGM's In particular
 - Efficient computation
 - Inter-operator optimizations
 - Intra-operator optimizations
 - Scalable computation
 - Parallel computation
- User Interface, applications, demo