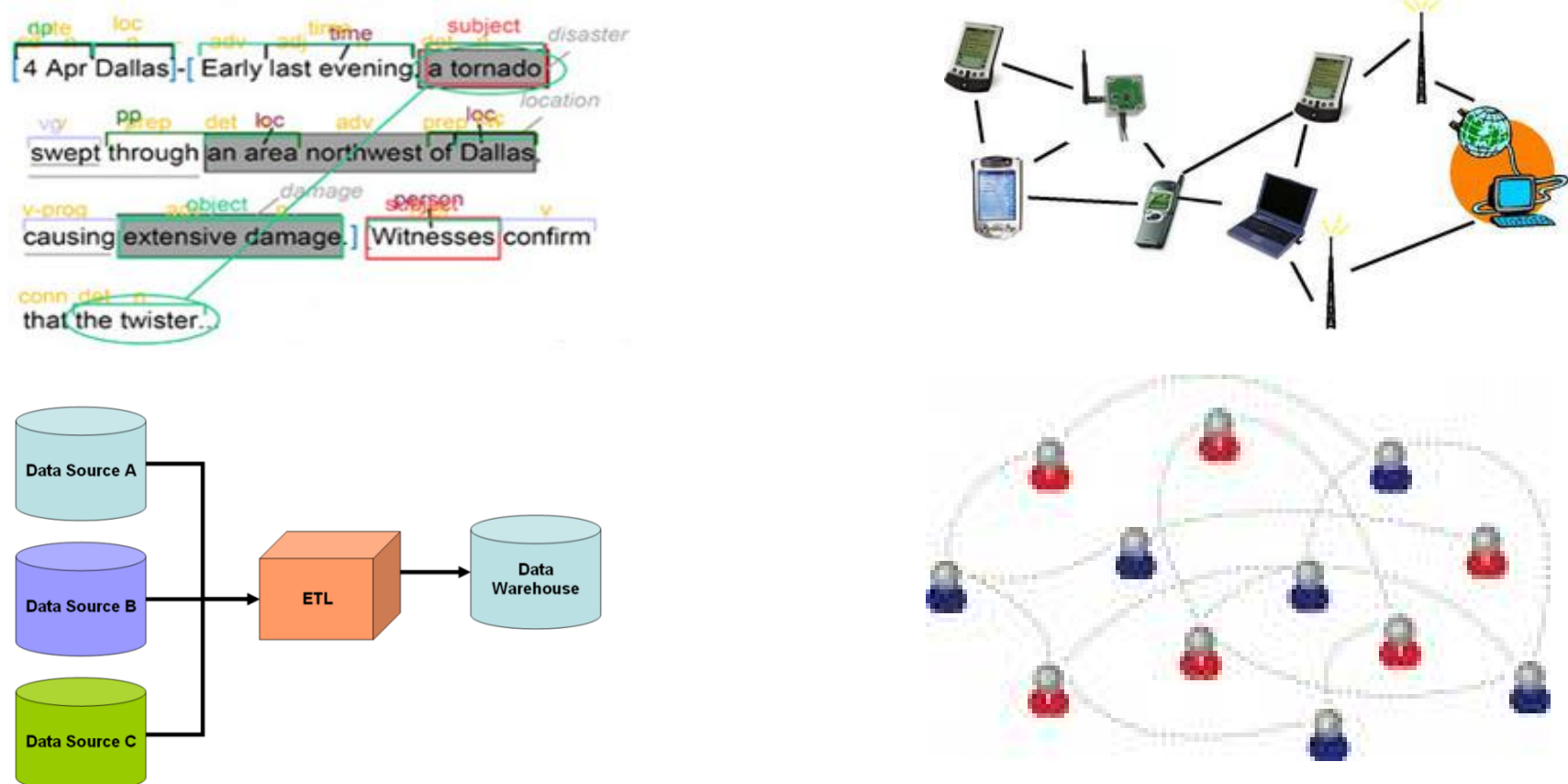


# BayesStore: Scalable Declarative Information Extraction

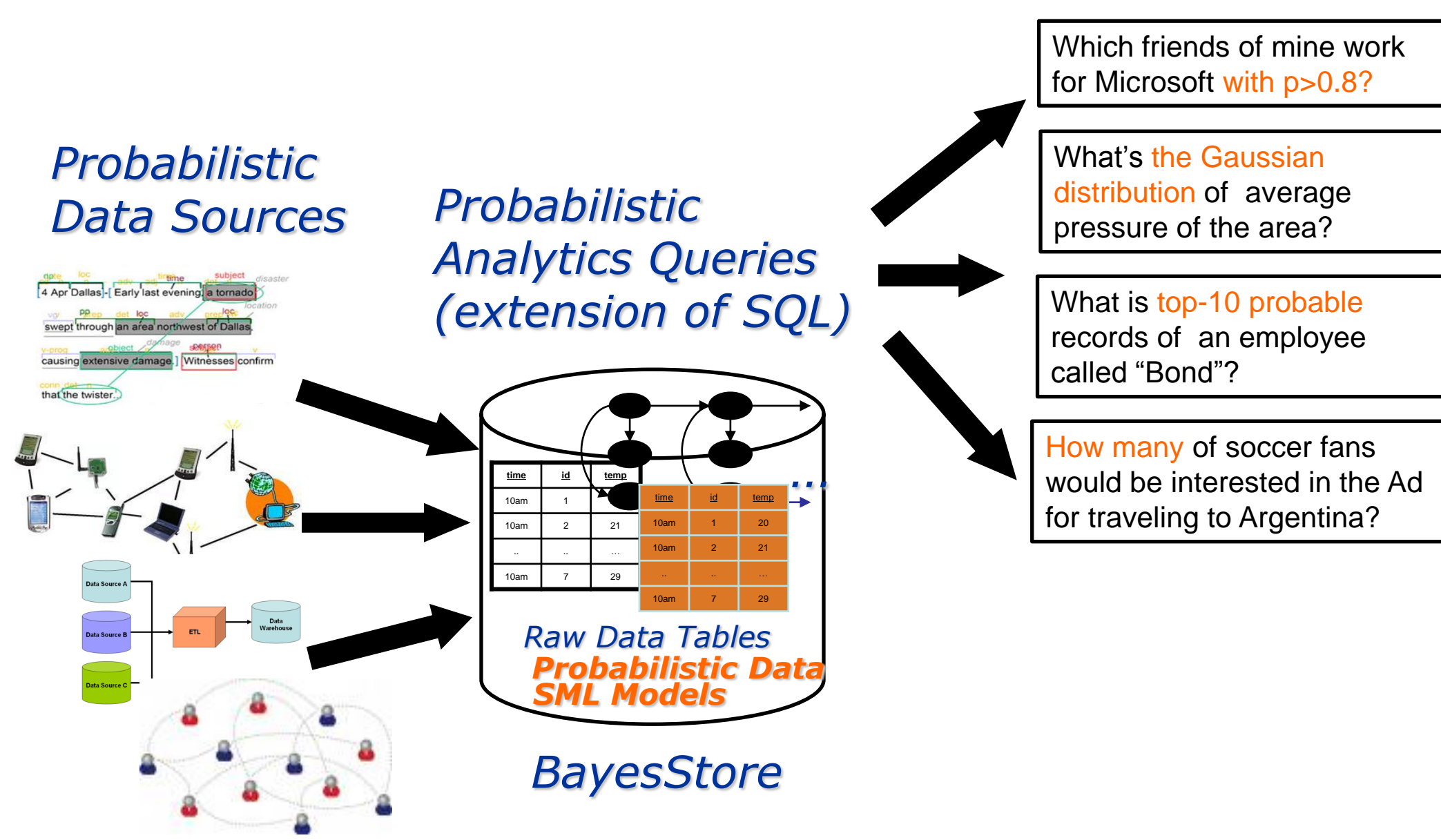


Daisy Z. Wang, Michael J. Franklin, Joseph M. Hellerstein, Minos Garofalakis

## Probabilistic Data Analytics



## BayesStore Approach vs. State-of-the-art

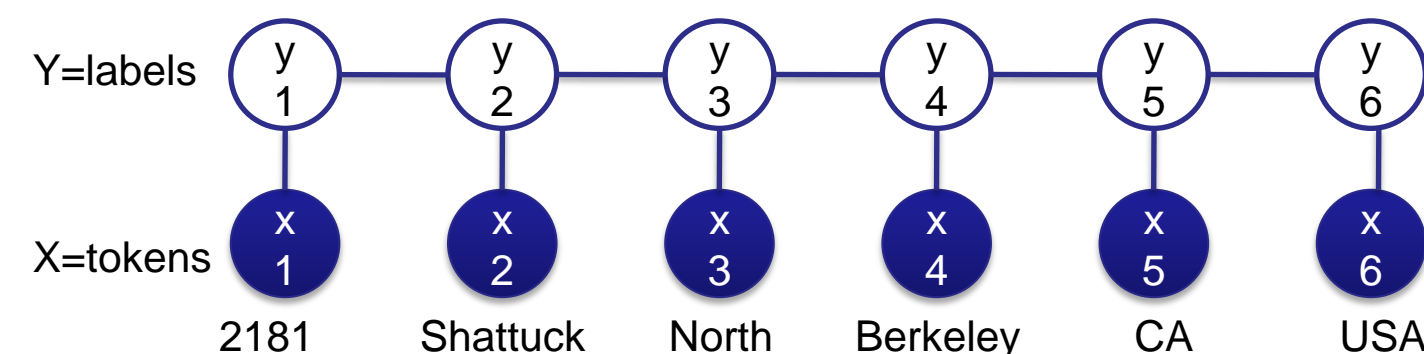


## Probabilistic Information Extraction – CRF Model and Viterbi Algorithm

Possible Extraction Worlds:

x	2181	Shattuck	North	Berkeley	CA	USA	(0.6)
y1	apt. num	street name	city	city	state	country	(0.1)
y2	apt. num	street name	street name	city	state	country	(0.1)

CRF Model:



Viterbi Dynamic Programming Algorithm:

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

Dynamic Programming V matrix:

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

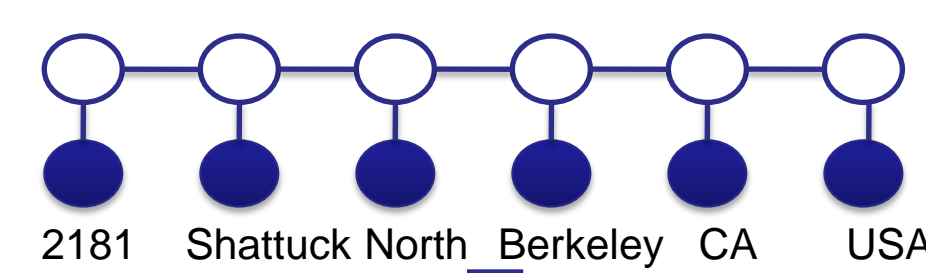
## Text Data and CRF Representation in BayesStore

- Text Data – Token Table: Inverted file, one token per row
- CRF – Factor Table: one <token, prevLabel, label> triple per row

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."

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

Token Table

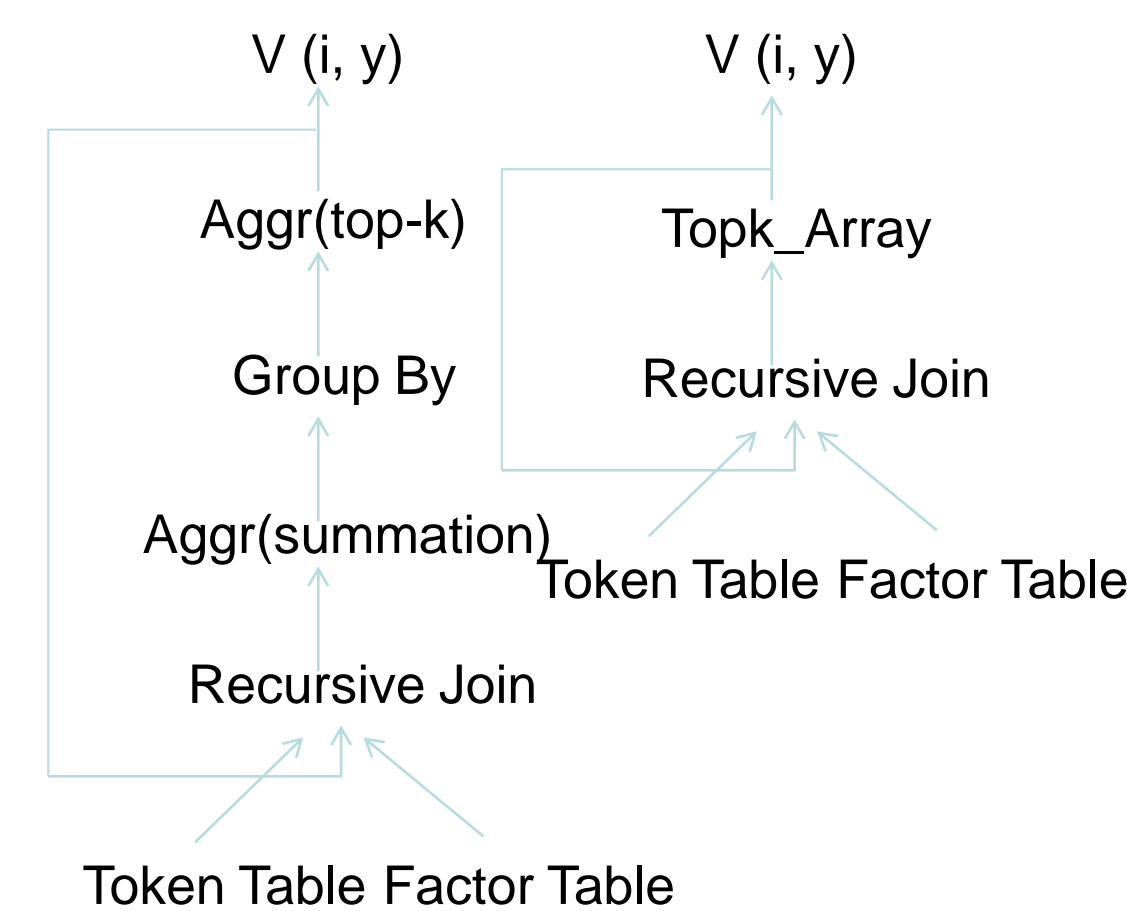


token	prevLabel	label	score
2181(DIGIT)	null	street num	22
2181(DIGIT)	null	street name	5
...	..	..	..
Berkeley	streetname	street name	10
Berkeley	streetname	city	25
..	..	..	..

Factor Table

## Viterbi Implementation in SQL

- ViterbiPerDoc (ViterbiAllDoc)
  - 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



## Entity Table and Two Query Families

Address (EntityTbl1)

strID	apt. num <sup>a</sup>	street num <sup>a</sup>	street name <sup>a</sup>	city <sup>a</sup>	state <sup>a</sup>	country <sup>a</sup>
1	null	2181	Shattuck	North Berkeley	CA	USA
2	12B	331	Fillmore St.	Seattle	WA	USA
3	224B	null	Ford South St.	Louis	MO	USA

Company (EntityTbl2)

strID	company name <sup>a</sup>	city <sup>a</sup>	state <sup>a</sup>
1	Google	Mountain View	CA
2	Yahoo!	Santa Clara	CA
3	Microsoft	Seattle	WA

Query Family 1: (SPJ-over-ML) SPJ over Maximum-likelihood (ML) Queries

```
CREATE VIEW entityTbl1-ML as
SELECT *, rank() OVER (ORDER BY prob(*) DESC) r
FROM entityTbl1
WHERE r = 1;
```

Query Family 2: (Probabilistic SPJ) Top-k over Probabilistic SPJ Queries

```
SELECT *, rank() OVER (ORDER BY prob(*) DESC) r
FROM SQLQuery
WHERE r <= k [ AND prob(*) > threshold ]
```

## Evaluation1: [Runtime Efficiency] SQL Implementations of Viterbi

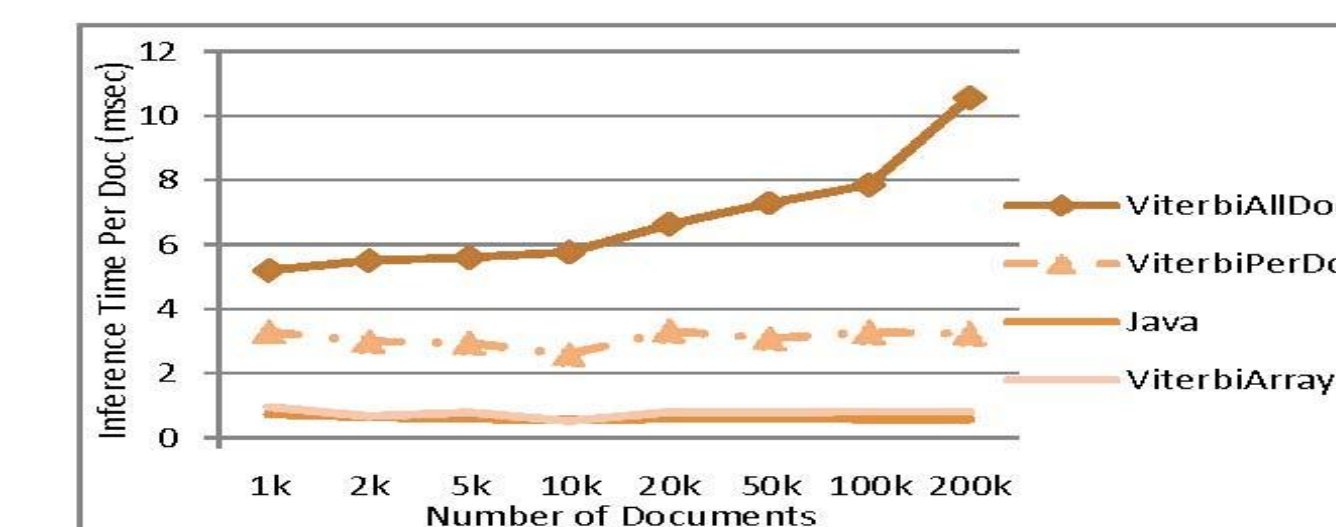
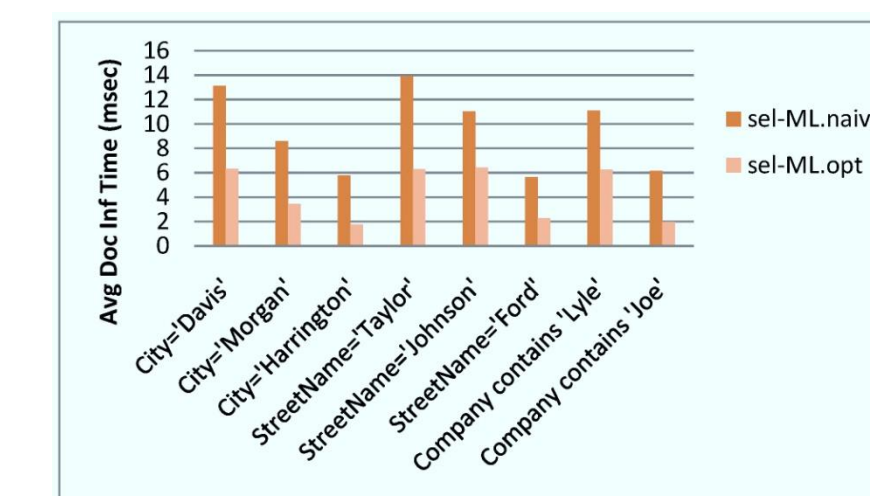


Figure 6: Average inference time (msec) for a single document for different implementations of the Viterbi algorithm.

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

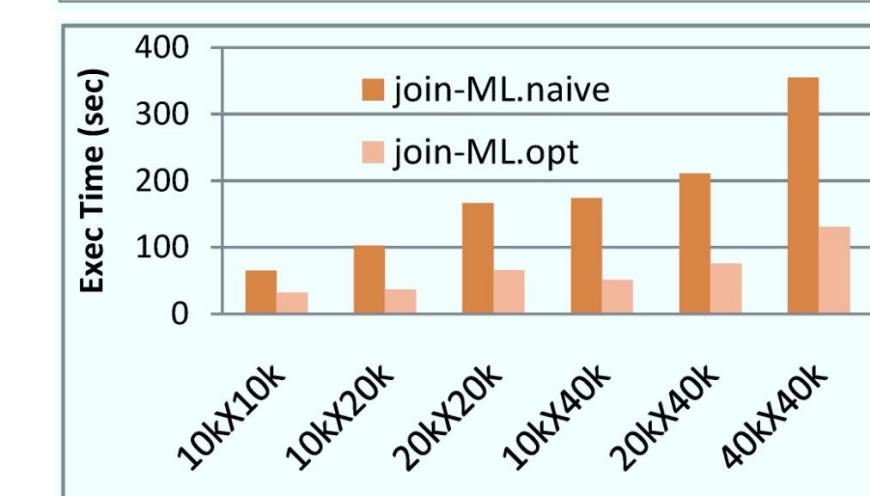
Figure 7: Average inference time per document (msec) for different Viterbi implementations on address and bib dataset.

## Evaluation 2: [Runtime Efficiency] Optimized SPJ-over-ML Queries



Select-over-ML: naive vs. opt

Figure 1: Performance comparison between sel-ML.naive and sel-ML.opt with difference selection conditions.



Join-over-ML: naive vs. opt

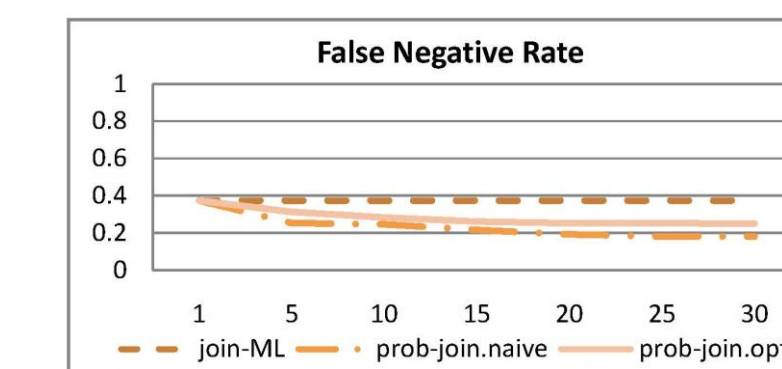
Figure 2: Performance comparison between join-ML.naive and join-ML.opt with different input sizes.

## Evaluation 3: [Answer Quality] Probabilistic SPJ Queries

(False -)	company	firstname	lastname	jobtitle	department
sel-ML	0.074	0.012	0.036	0.102	0.286
prob-sel	0.014	0	0.006	0.037	0.079
(False +)	company	firstname	lastname	jobtitle	department
sel-ML	0.010	0	0	0.010	0
prob-sel	0.009	0.006	0.006	0.010	0

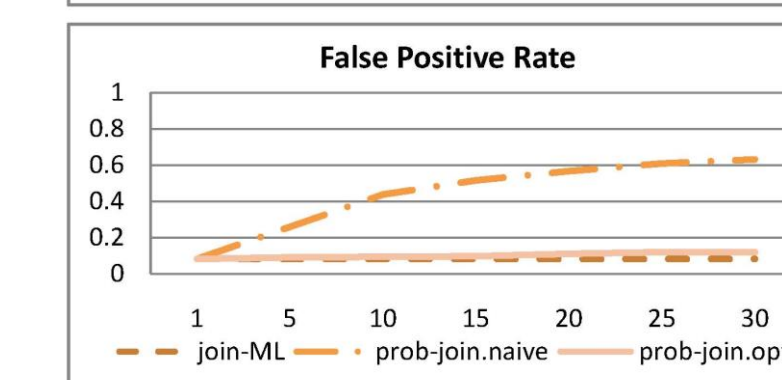
Probabilistic Select vs. Select-over-ML

Figure 1: False negative, false positive rates comparison between sel-ML and prob-sel queries with different selection conditions.



Probabilistic Join vs. Join-over-ML

Figure 2,3: False negative, false positive rates comparison between join-ML and prob-join queries with different k values.



## Future Work

- Parallelization based on HadoopDB
- Extensibility and Abstractions
- Other Applications