Learned Cardinalities: Estimating Correlated Joins with Deep Learning

Cardinality estimation problem

what it is + why is it hard

Key ideas

Discussion

CS294 AI-Sys
Presented by: Zongheng Yang
zongheng@berkeley.edu

Cardinality Estimation

Single-table

```
SELECT * FROM sal
WHERE sal.position = ‘Manager I’
AND   sal.salary >100,000
```

emp_id	position	country
1	Manager II	USA
2	Engineer I	CAN
3	Engineer II	USA
4	…	..

sal_id	position	salary
1	Manager I	120000.00
2	Manager II	150000.00
3	Engineer I	78000.00
4	Engineer II	91000.00

tax_id	country	rate
1	USA	0.32
2	CAN	0.45
3	CHN	0.17
4	…	…

Cardinality Estimation

Single-table

```
SELECT * FROM sal
WHERE sal.position = ‘Manager I’
AND   sal.salary >100,000
```

Likely! (correlation)

emp_id	position	country
1	Manager II	USA
2	Engineer I	CAN
3	Engineer II	USA
4	…	..

sal_id	position	salary
1	Manager I	120000.00
2	Manager II	150000.00
3	Engineer I	78000.00
4	Engineer II	91000.00

tax_id	country	rate
1	USA	0.32
2	CAN	0.45
3	CHN	0.17
4	…	…

Cardinality Estimation

Single-table

```
SELECT * FROM sal
WHERE sal.position = 'Manager I'
AND   sal.salary >100,000
```

Likely! (correlation)

```
SELECT * FROM twitter_graph
WHERE following = 'Michael Jordan'
```

Most! (uniformity)

```
SELECT * FROM cars
WHERE make = 'Honda'
AND model = 'Jetta'
```

Anti-correlation!

emp_id	position	country
1	Manager II	USA
2	Engineer I	CAN
3	Engineer II	USA
4	…	..

sal_id	position	salary
1	Manager I	120000.00
2	Manager II	150000.00
3	Engineer I	78000.00
4	Engineer II	91000.00

tax_id	country	rate
1	USA	0.32
2	CAN	0.45
3	CHN	0.17
4	…	…

Cardinality Estimation

Single-table

```
SELECT * FROM sal
WHERE sal.position = 'Manager I'
AND   sal.salary >100,000
```

Likely! (correlation)

```
SELECT * FROM twitter_graph
WHERE following = 'Michael Jordan'
```

Most! (uniformity)

```
SELECT * FROM cars
WHERE make = 'Honda'
AND model = 'Jetta'
```

Anti-correlation!

```
Reduction(query) = R(pred 1) * R(pred 2)
Reduction(col=val) = 1 / num_distinct(col)
```

emp_id	position	country
1	Manager II	USA
2	Engineer I	CAN
3	Engineer II	USA
4	…	..

sal_id	position	salary
1	Manager I	120000.00
2	Manager II	150000.00
3	Engineer I	78000.00
4	Engineer II	91000.00

tax_id	country	rate
1	USA	0.32
2	CAN	0.45
3	CHN	0.17
4	…	…

Cardinality Estimation

Joins

```
SELECT * FROM emp, sal
WHERE emp.position = 'Manager I'
AND   sal.salary >100,000
```

emp_id	position	country
1	Manager II	USA
2	Engineer I	CAN
3	Engineer II	USA
4	…	..

sal_id	position	salary
1	Manager I	120000.00
2	Manager II	150000.00
3	Engineer I	78000.00
4	Engineer II	91000.00

tax_id	country	rate
1	USA	0.32
2	CAN	0.45
3	CHN	0.17
4	…	…

Cardinality Estimation

Joins

```
SELECT * FROM emp, sal
WHERE emp.position = 'Manager I'
AND   sal.salary >100,000
```

"correlated joins"

emp_id	position	country
1	Manager II	USA
2	Engineer I	CAN
3	Engineer II	USA
4	…	..

sal_id	position	salary
1	Manager I	120000.00
2	Manager II	150000.00
3	Engineer I	78000.00
4	Engineer II	91000.00

tax_id	country	rate
1	USA	0.32
2	CAN	0.45
3	CHN	0.17
4	…	…

Cardinality Estimation

Joins

```
SELECT * FROM emp, sal
WHERE emp.position = 'Manager I'
AND   sal.salary >100,000
```

"correlated joins"

```
Reduction(join) = 1 / max {
    Cardinality("emp where emp.pos = Mgr1"),
    Cardinality("sal where sal.sal > 100K")
}
```

emp_id	position	country
1	Manager II	USA
2	Engineer I	CAN
3	Engineer II	USA
4	…	..

sal_id	position	salary
1	Manager I	120000.00
2	Manager II	150000.00
3	Engineer I	78000.00
4	Engineer II	91000.00

tax_id	country	rate
1	USA	0.32
2	CAN	0.45
3	CHN	0.17
4	…	…

How bad is it?

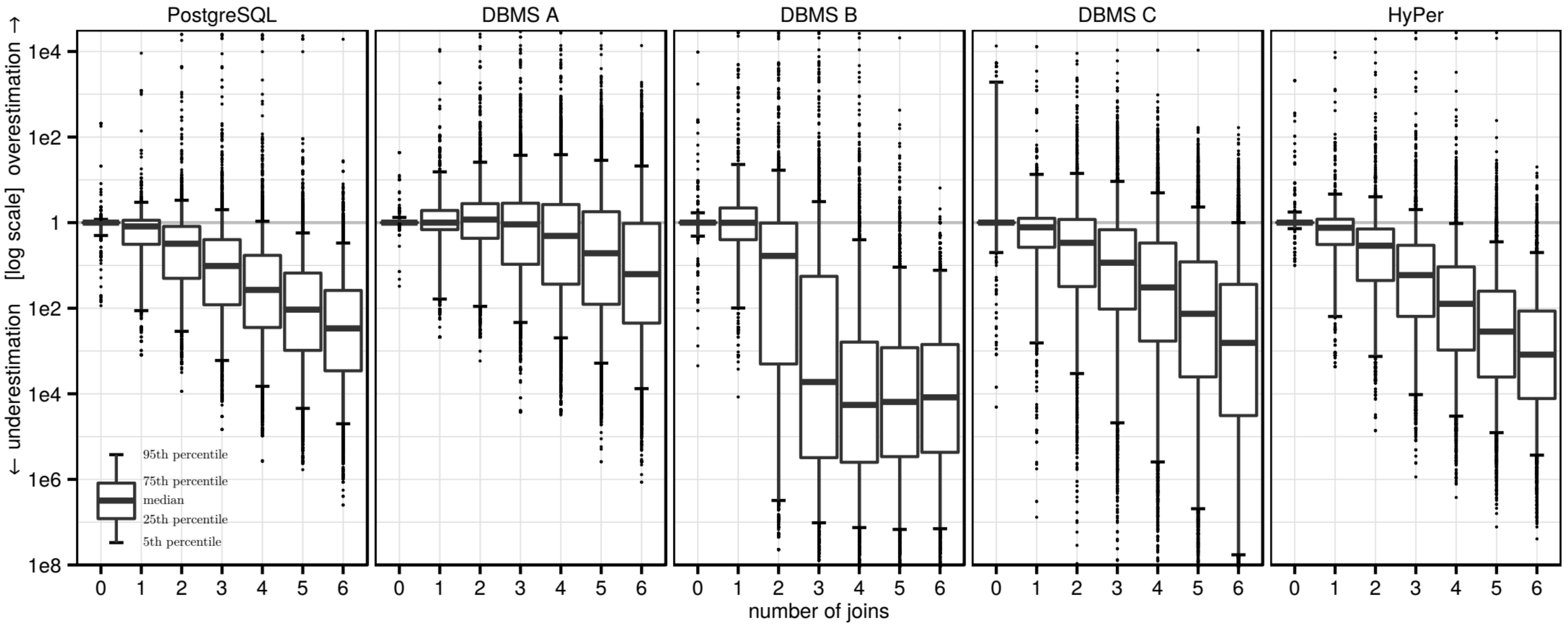


Figure 3: Quality of cardinality estimates for multi-join queries in comparison with the true cardinalities. Each boxplot summarizes the error distribution of all subexpressions with a particular size (over all queries in the workload)

VLDB'15, Leis et al., How Good Are Query Optimizers, Really?

How bad is it?

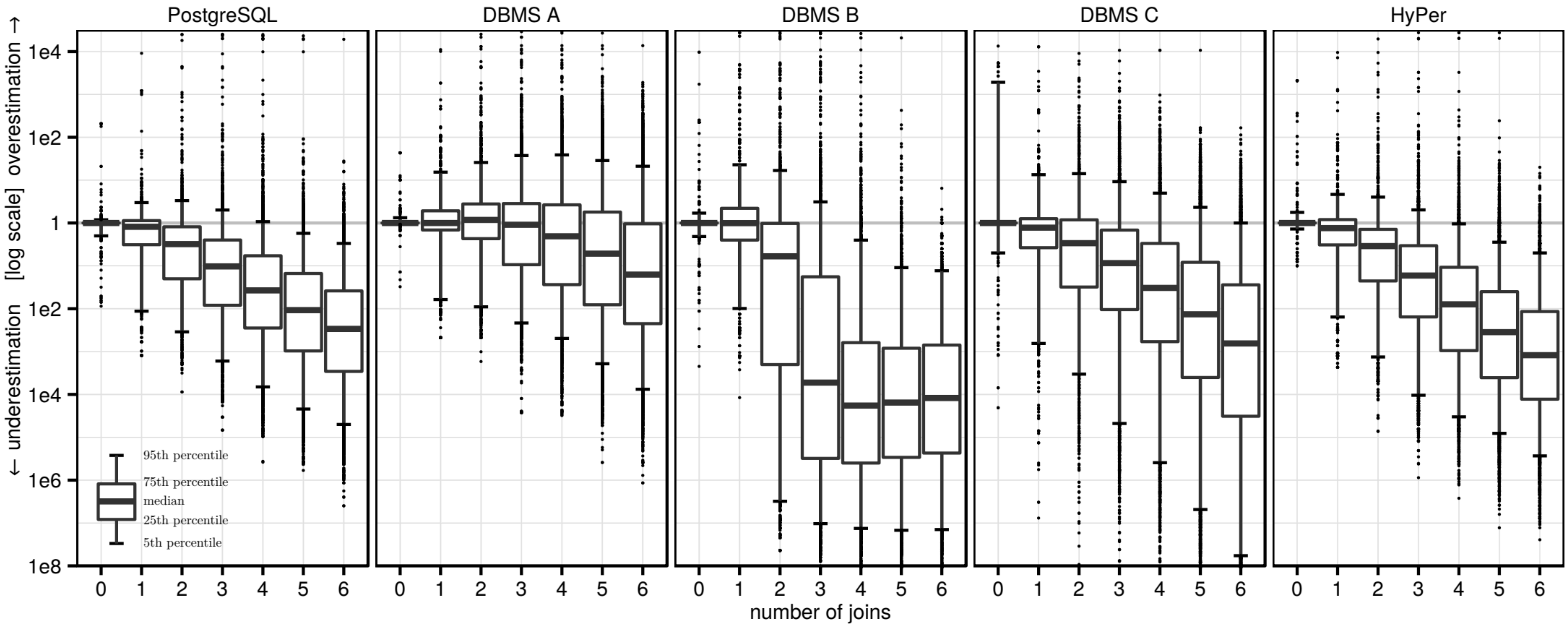


Figure 3: Quality ... **plot summarizes**
the error distribu...

For 6-way joins: median 100x off, outliers up to 10^8x off

VLDB'15, Leis et al., How Good Are Query Optimizers, Really?

Key Ideas

- Recall: uniformity & independence assumptions are bad

Key Ideas

- Recall: uniformity & independence assumptions are bad
- What if we give a model:

Features	Labels (cardinality)
following = Jordan	1 million (likely)
following = Nadorj	10 (unlikely)
age < 20 && salary > 100K	1K (unlikely)
age > 30 && salary > 100K	100K (likely)

Key Ideas

- Recall: uniformity & independence assumptions are bad
- What if we give a model:

Features	Labels (cardinality)
following = Jordan	1 million (likely)
following = Nadorj	10 (unlikely)
age < 20 && salary > 100K	1K (unlikely)
age > 30 && salary > 100K	100K (likely)

- It should then learn to fix unif./indep. assumptions!

Key Ideas

- This is exactly what they did!

Key Ideas

- This is exactly what they did!

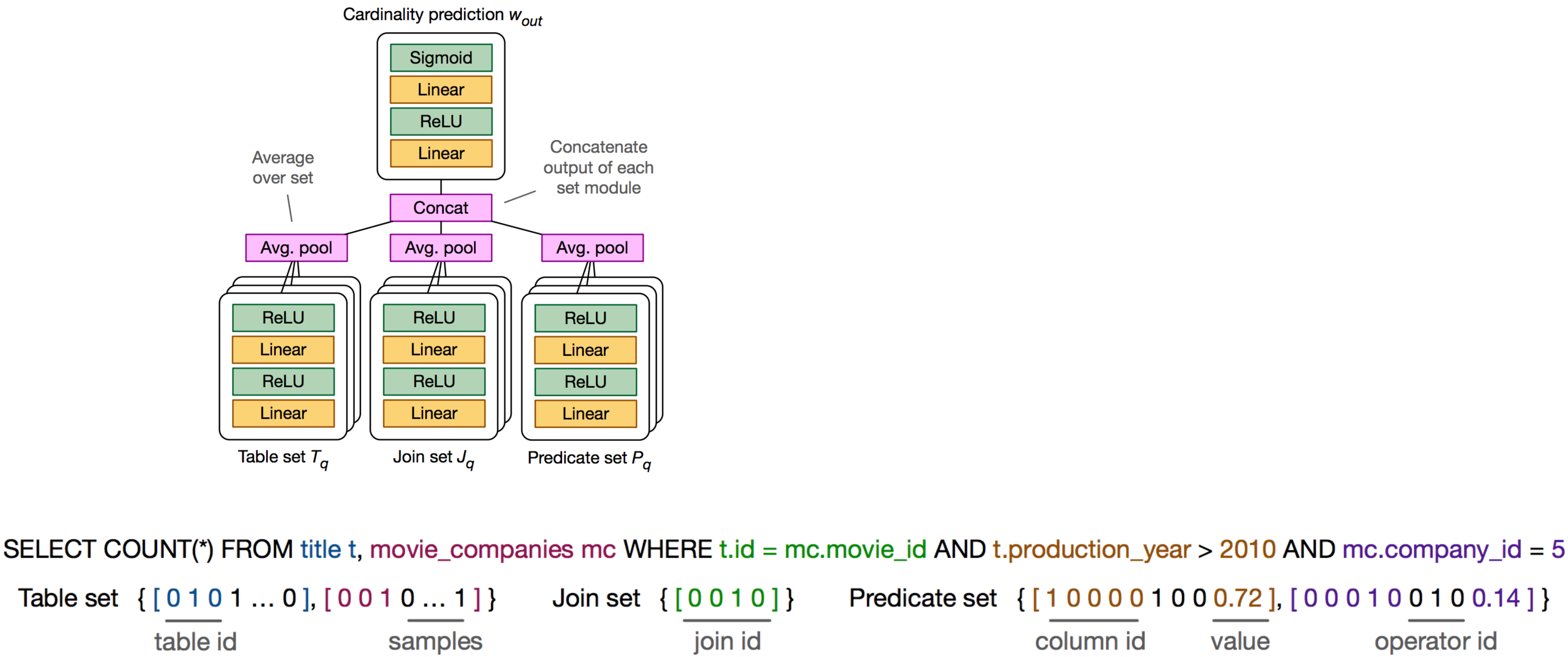
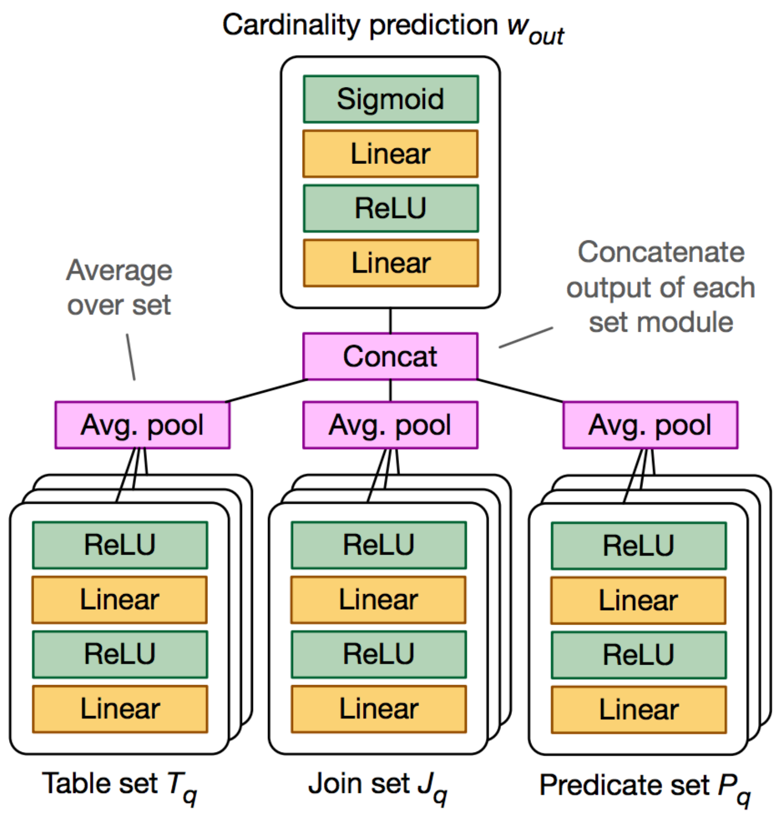


Figure 2: Query featurization as sets of feature vectors.

Key Ideas

- This is exactly what they did!



"Our query generator first **uniformly draws the number of joins** |Jq | (0 ≤ |Jq | ≤ 2) and then uniformly selects a table that is referenced by at least one table. For |Jq | > 0, it then **uniformly selects a new table** that can join with the current set of tables (initially only one), adds the corresponding join edge to the query and (overall) repeats this process |Jq | times. For each base table t in the query, it then **uniformly draws the number of predicates** |P t q | (0 ≤ |P t q | ≤ num non-key columns). For each predicate, it **uniformly draws the predicate type** (=, <, or >) and **selects a literal (an actual value) from the corresponding column.**"

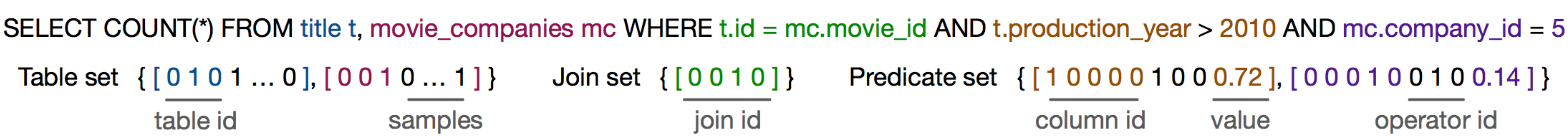


Figure 2: Query featurization as sets of feature vectors.

Assumptions

Assumptions

SELECT COUNT(*) FROM title t, movie_companies mc WHERE t.id = mc.movie_id AND t.production_year > 2010 AND mc.company_id = 5

Table set { [0 1 0 1 … 0], [0 0 1 0 … 1] }   Join set { [0 0 1 0] }   Predicate set { [1 0 0 0 0 1 0 0 0.72], [0 0 0 1 0 0 1 0 0.14] }

table id   samples   join id   column id   value   operator id

Figure 2: Query featurization as sets of feature vectors.

Assumptions

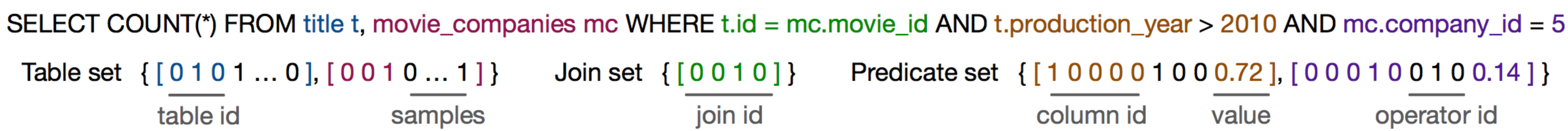


Figure 2: Query featurization as sets of feature vectors.

- Assume
  - Static column range (2010 -> 0.72); no appends
  - Static DB schema (same set of tables, cols)
  - Training data MUST cover well desired queries
  - Quality depends on ACTUAL execution on a small sample from each table, at query time

Results

	median	90th	95th	99th	max	mean
PostgreSQL	7.93	164	1104	2912	3477	174
Random Samp.	11.5	198	4073	22748	23992	1046
IB Join Samp.	**1.59**	150	3198	14309	15775	590
MSCN	3.82	**78.4**	**362**	**927**	**1110**	**57.9**

Table 4: Estimation errors on the JOB-light workload.

Results

	median	90th	95th	99th	max	mean
PostgreSQL	7.93	164	1104	2912	3477	174
Random Samp.	11.5	198	4073	22748	23992	1046
IB Join Samp.	**1.59**	150	3198	14309	15775	590
MSCN	3.82	**78.4**	**362**	**927**	**1110**	**57.9**

Table 4: Estimation errors on the JOB-light workload.

Up to 4 joins (5 tables):

3x better than Postgres @max and @mean

What is actually learned?

What is actually learned?

```
Reduction(join) = 1 / max {
    Cardinality("emp where emp.pos = Mgr1"),
    Cardinality("sal where sal.sal > 100K")
}
```

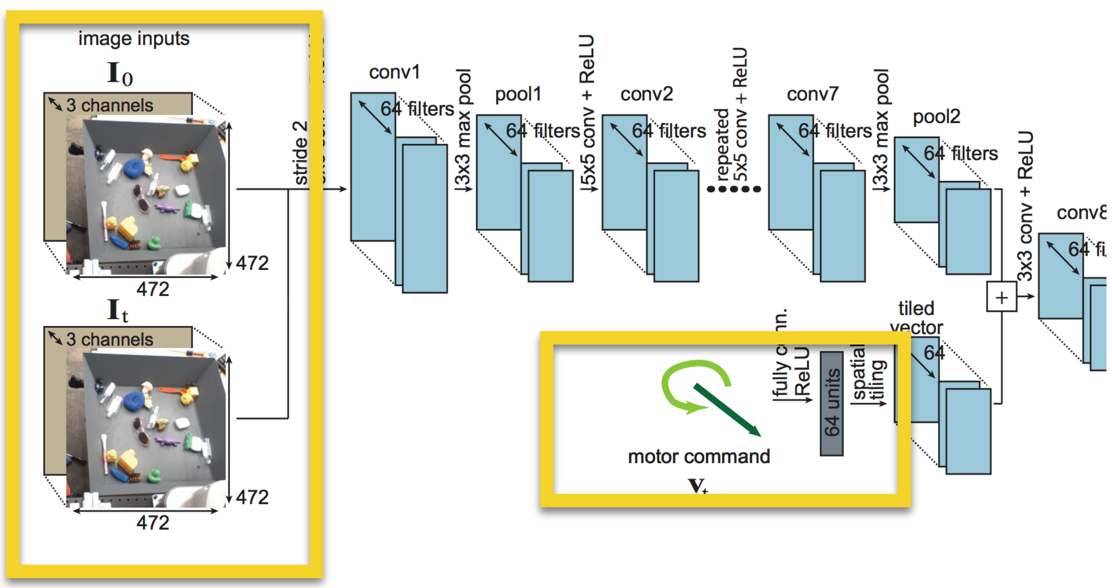
What is actually learned?

```
Reduction(join) = 1 / max {
    Cardinality("emp where emp.pos = Mgr1"),
    Cardinality("sal where sal.sal > 100K")
}
```

- My interpretation
  - It learns a *dampened* version of this formula per column/predicate combination
    - This "solves" correlation
  - Deep nets are great at capturing patterns

Discussion

- Vision and Control - is it useful to have "vision" in understanding databases' data?



Levine et al., Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection

- Tree/graph neural nets needed (or even helpful) here?

- Do learning solutions have a place for "easy" cases? (How to afford data/training/operational costs?)