Table 3: Overall performance of CardEst algorithms.

Category	Method	JOB-LIGHT End-to-End Time	JOB-LIGHT Exec. + Plan Time	JOB-LIGHT Improvement	STATS-CEB End-to-End Time	STATS-CEB Exec. + Plan Time	STATS-CEB Improvement
Baseline	PostgreSQL	3.67h	3.67h + 3s	0.0%	11.34h	11.34h + 25s	0.0%
	TrueCard	**3.15h**	**3.15h + 3s**	**14.2%**	**5.69h**	**5.69h + 25s**	**49.8%**
Traditional	MultiHist	3.92h	3.92h + 30s	−6.8%	14.55h	14.53h + 79s	−28.3%
	UniSample	4.87h	4.84h + 96s	−32.6%	$> 25h$	−−	−−
	WJSample	4.15h	4.15h + 23s	−13.1%	19.86h	19.85h + 45s	−75.0%
	PessEst	3.38h	3.38h + 11s	7.9%	6.10h	6.10h + 43s	46.2%
Query-driven	MSCN	3.50h	3.50h + 12s	4.6%	8.13h	8.11h + 46s	28.3%
	LW-XGB	4.31h	4.31h + 8s	−17.4%	$> 25h$	−−	−−
	LW-NN	3.63h	3.63h + 9s	1.1%	11.33h	11.33h + 34s	0.0%
	UAE-Q	3.65h	3.55h+356s	−1.9%	11.21h	11.03h+645s	1.1%
Data-driven	NeuroCard^E	3.41h	3.29h + 423s	6.8%	12.05h	11.85h + 709s	−6.2%
	BayesCard	3.18h	3.18h + 10s	13.3%	7.16h	7.15h + 35s	36.9%
	DeepDB	3.29h	3.28h + 33s	10.3%	6.51h	6.46h + 168s	42.6%
	FLAT	3.21h	3.21h + 15s	12.9%	5.92h	5.80h + 437s	47.8%
Query + Data	UAE	3.71h	3.60h + 412s	−2.7%	11.65h	11.46h + 710s	−0.02%

Table 4: End-to-end time improvement ratio of CardEst algorithms on queries with different number of join tables.

# tables	# queries	PessEst	MSCN	BayesCard	DeepDB	FLAT	TrueCard
2 − 3	38	2.62%	2.04%	2.07%	1.98%	2.48%	**3.66%**
4	50	53.1%	−12.3%	55.8%	48.0%	55.7%	**55.9%**
5	28	31.7%	29.8%	36.55%	32.90%	35.4%	**37.0%**
6 − 8	34	29.6%	−4.06%	2.51%	26.3%	32.0%	**34.6%**

This complicated distribution of join keys makes NeuroCard^E's full outer join sample less effective. Therefore, NeuroCard^E can hardly capture the correct data distributions especially for join tables with small cardinalities. Specifically, we find that for queries on the joins of a small set of tables, NeuroCard^E's prediction deviates significantly from the true cardinality because its training sample does not contain much not-null tuples for this particular set of join tables.

All other three data-driven CardEst methods can significantly outperform the PostgreSQL baseline because their models are not constructed on the full outer join of all tables. Specifically, they all use the "divide and conquer" idea to divide the large join schema into several smaller subsets with each representing a join of multiple tables. In this way, they can capture the rich correlation within each subset of tables; simultaneously, they avoid constructing the full outer join of all tables by assuming some independence between tables with low correlations. Then, BayesCard, DeepDB, and FLAT build a model (BN, SPN, and FSPN respectively) to represent the distribution of the corresponding small part. This approach solves the drawback of NeuroCard^E, yields relatively accurate estimation, and produces effective query execution plans. Among them, FLAT achieves the best performance (47.8% improvement), which is very close to the improvement 49.8% for TrueCard. It can outperform DeepDB mostly because the STATS dataset is highly correlated, so the FSPN in FLAT has a more accurate representation of the data distribution than the SPN in DeepDB. On the other hand, BayesCard has an even more accurate representation of data distribution and yields the best end-to-end time for most queries in STATS-CEB. It does not outperform FLAT most because of one extremely long-run query, which we will study in detail in Section 5.2.

5.2 Analysis of Different Query Settings

In this section, we further examine to what extent the CardEst methods improve over PostgreSQL on various query types, i.e. different number of join tables (#$tables$) and different intervals of true cardinalities. Since JOB-LIGHT workload does not contain queries with very diverse types and the ML-based data-driven methods do not show significant difference on these queries, we only investigate queries on STATS-CEB. Worth noticing that we only examine the methods with clear improvements over PostgreSQL on STATS-CEB: MSCN, BayesCard, DeepDB, and FLAT.

Number of Join Tables: Table 4 shows performance improvement of different ML-based methods over the PostgreSQL baseline and we derive the following observation:

O4: The improvement gaps between these methods and the performance of TrueCard increase with the number of join tables. Specifically, the BayesCard achieves near-optimal improvement for queries joining no more than 5 tables, but it barely has much improvement for queries joining 6 tables and more. This observation suggests that the estimation qualities of these SOTA methods decline for queries joining more tables. In fact, the fanout join estimation approach adopted by all these methods sacrifices accuracy for efficiency by assuming some tables are independent of others. This estimation error may accumulate for queries joining a large number of tables, leading to sub-optimal query plans.

Size of Cardinality: We choose Q57 (in Figure 2) of STATS-CEB as a representative query to study the effect of estimation accuracy w.r.t. different cardinalities and investigate when a CardEst method could go wrong. The execution time of Q57 for TrueCard and FLAT is 1.90h and 1.92h, while the time for BayesCard is 3.23h. We derive two important observations from this query, which are verified to be generalizable to other queries in JOB-LIGHT and STATS-CEB.

O5: Accurate estimation of (sub-plan) queries with large cardinalities is sometimes more important than the small ones. When choosing the join method in the root node of execution plans for Q57, BayesCard underestimates the final join size and chooses