A MORE DETAILS ABOUT OFFLINE DATA GENERATION

A.1 Profile of The Labeled Tasks

Table 4 displays the statistics of the labeled tasks. In the process of preparing the tuning tasks, we considered tuning the top-n workload-specific or generally important knobs (with n ranging from 3 to 197 for MySQL). This led to the creation of 390 distinct spaces (2×195). The labeled tasks were thoughtfully selected by active learning to encompass informative tuning conditions from a large number of candidates.

A.2 Analysis of Workload Similarity

To evaluate workload similarity, we gathered extensive observations for the training and testing workload through LHS-based configuration sampling and the collection of tuning histories. We quantify their similarity using the concordant ranking ratio, and the results are presented in Figure 14, in which SYSBENCH RW, SYSBENCH RO, SYSBENCH WO, and Twitter is the testing workloads adopted in our experiments. We further analyze the overlap of effective ranges between workloads, which forms the basis for our space transfer strategy. The effective range represents the minimal range that encompasses all potential configurations with performance exceeding a given standard f^b. Figure 15 presents data on average effective range sizes (diagonal elements) and their overlaps between workloads (off-diagonal elements). The effective range values are normalized against the default range, yielding values between zero and one. We observe a positive relationship between the overlap of effective ranges (Figure 15) and task similarities (Figure 14). This indicates that more similar tasks exhibit a greater overlap in their effective ranges. Figure 15 also illustrates that a larger value of f^b corresponds to a smaller effective range. These observation aligns with OpAdviser's strategy of controlling the size of effective range based on task similarity: when a source workload is more similar to the target, OpAdviser prunes unpromising regions based on the source observations more aggressively. Conversely, when the source workload is less similar to the target, it applies a more gentle space pruning.

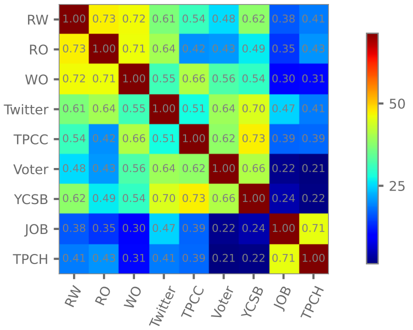


Figure 14: Similarity Between Different Workloads.

Table 4: The Statistics of The Labeled Tasks.

Task ID	Workload	Knob Number	Categorical Ratio(%)	Space Size (log 10)
1	Twitter	3	0.0	16.6
2	TPCH	50	32.0	164.0
3	Sysbench RO	118	50.8	308.6
4	JOB	5	0.0	36.2
5	Sysbench RO	29	10.3	133.8
6	Sysbench RW	11	18.2	60.3
7	Twitter	28	7.1	168.5
8	Sysbench RW	133	51.9	350.8
9	YCSB	174	42.5	609.7
10	Sysbench WO	49	28.6	184.9
11	Sysbench WO	96	44.8	313.2
12	Twitter	47	36.2	171.0
13	Voter	80	42.5	287.5
14	TPC-C	7	0.0	27.2
15	Sysbench RO	12	8.3	79.5
16	TPC-C	138	50.7	382.7
17	JOB	11	9.1	77.5
18	Twitter	6	16.7	34.1
19	Twitter	45	37.8	146.6
20	TPCH	139	38.1	556.7
21	Sysbench RO	22	4.5	148.4
22	Sysbench WO	74	32.4	280.9
23	Voter	67	50.7	172.7
24	Sysbench WO	40	7.5	226.9
25	Sysbench RO	8	10.2	65.7
26	Twitter	125	38.4	532.7
27	YCSB	84	46.4	299.3
28	Twitter	52	36.5	182.6
29	JOB	74	32.4	280.9
30	YCSB	7	14.3	36.6
31	YCSB	20	5.0	124.0
32	TPC-C	10	10.0	68.5
33	JOB	84	53.6	213.7
34	Sysbench RO	5	20.0	31.1
35	YCSB	57	22.8	253.5
36	JOB	86	53.5	223.7
37	Sysbench WO	96	44.8	289.3
38	JOB	19	5.3	115.0
39	YCSB	26	7.7	159.8
40	YCSB	144	38.9	581.0
41	TPC-C	190	38.9	704.8
42	Sysbench RO	18	5.6	104.0
43	JOB	26	15.4	129.1
44	TPCH	40	22.5	154.4
45	TPC-C	105	42.9	377.9
46	Sysbench WO	170	43.5	570.3
47	Sysbench RW	7	28.6	35.9
48	TPC-C	94	43.6	288.7

A.3 Retraining of Meta-ranker

The offline collection is recommended to repeat when the DBMS changes, because different systems have varying numbers and compositions of knobs, as seen in the contrast between MySQL and Postgres. However, once the meta-ranker is trained for a specific

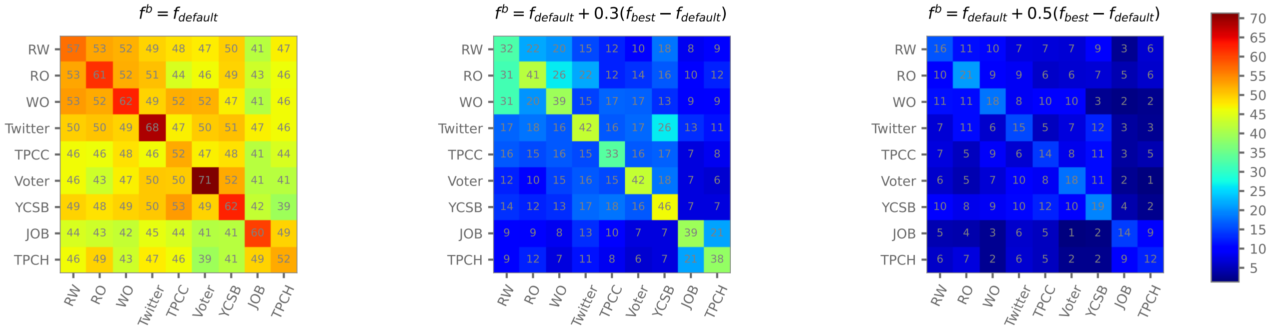


Figure 15: Visualization of The Average Effective Range per Knob. Diagonal elements indicates the average effective range for individual workloads and off-diagonal elements signify the average overlap of effective ranges between workloads.

DBMS, the need of further offline data collection for the DBMS becomes minimal, except in cases of significant system updates that, for example, deprecate or introduce a larger percentage of knobs. This is because that the offline data collection primarily serves to train the meta-ranker within the optimizer selection module, which replaces the heuristics, such as "DDPG performs well in large search spaces" or "start with GA before switching to DDPG". Manually determining which heuristics to adopt is complex, involving establishing suitable decision thresholds, such as at which size DDPG is preferred over SMAC or the iteration count for transitioning from GA to DDPG. However, the meta-ranker learns the optimal selection decisions automatically in a data-driven way. In our experiments, the meta-ranker has an accuracy of 0.84 for predicting the best optimizer in leave-one-out validation, demonstrating its generality for unseen tasks. Even when we exclude the response surface feature from training (considering only the feature of search space and tuning iterations), the meta-ranker still achieves an accuracy of 0.76. This indicates that it captures underlying patterns beyond workload types.

B MORE EXPERIMENTS

B.1 Comparison of Different Similarity Measurements

We compare OpAdviser's relative ranking approach with OtterTune's workload mapping [7]. OtterTune's workload mapping identifies the most similar workload by analyzing the internal runtime metrics of the DBMS, such as MySQL's counters for pages read or written to disk. It calculates the Euclidean distance between the target metrics and the predicted metrics from the sources, selecting the workload with the minimal distance as the most similar.

Figure 16 presents the performance of OpAdviser using relative ranking and workload mapping as similarity measurements, respectively. We observe that relative ranking yields superior performance. There are two primary issues with workload mapping. First, it relies on the absolute distance between DBMS metrics, which can vary significantly in scale across different hardware environments. Consequently, OtterTune restricts knowledge transfer within the same hardware type [7]. In contrast, OpAdviser measures similarity based on the relative ranking of configurations rather than their absolute performance values, effectively avoiding this problem. Second, the similarity of DBMS metrics does not always correspond to the similarity of the optimal region, which is more directly quantified by the ratio of concordant ranking pairs between two tasks.

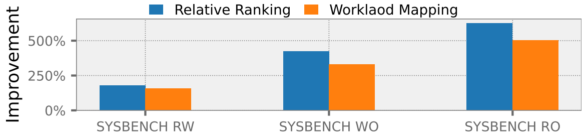
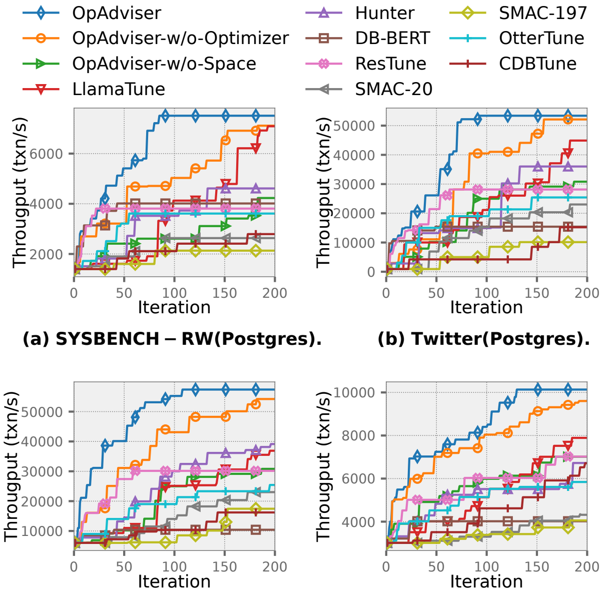


Figure 16: Comparison of Different Similarity Measurement.

B.2 Evaluation on More DBMSs

We expand our evaluation to include an end-to-end comparison on Postgres and MongoDB. These evaluations are carried out on a cloud instance with an 8-core CPU and 16 GB of memory, where Postgres has 169 configurable knobs and MongoDB has 72 configurable knobs. Compared with MySQL, they have completely different configurable knobs including different meanings, types, names and value ranges. Besides, MongoDB is a NoSQL databases, which is based on key-value data structure. We employ publicly available benchmarks to collect training data for the meta-ranker on an instance with 8 cores CPU and 8 GB memory. Specifically, we label 35 tasks with OLTP-Bench [15] for Postgres and 20 tasks with NoSQLBench [4] for MongoDB, following the procedure outlined in Section 6.2. Figure 17 presents the results. OpAdviser demonstrates robust performance on Postgres and MongoDB, consistent with its performance on MySQL. This highlights two key points: Firstly, OpAdviser's design emphasizes strong scalability in database tuning, adaptable to various DBMSs. Secondly, many other DBMSs feature broad default knob ranges, offering OpAdviser opportunities to expedite tuning by eliminating unpromising configuration spaces.

The default knob ranges for a DBMS are intentionally designed to cover all potential workload scenarios, rather than being customized for a specific workload. This assumption remains valid for other database systems as well. Figure 18 visually depicts the space



(a) SYSBENCH – RW(Postgres). **(b) Twitter(Postgres).**

(c) SYSBENCH – RO(MongoDB). **(d) YCSB – RW(MongoDB).**

Figure 17: Evaluation on Different Databases.

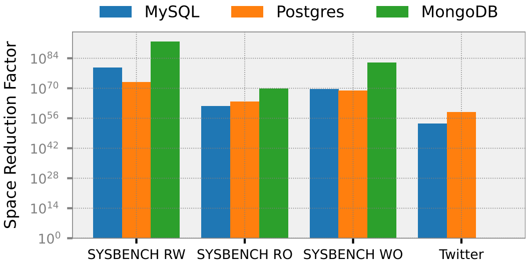


Figure 18: Space Reduction Factor in Different Database.

reduction factor ($\frac{Sizeof(default\ space)}{Sizeof(effective\ space)}$) achieved through the utilization of effective ranges. The results for Twitter in MongoDB is not shown, since the OLTPBench currently does not support MongoDB. We observe that the utilization of effective ranges leads to a significant reduction in the size of search space. It is important to note that in the above discussion, we employ the default performance as the performance standard (denoted as f_b) to define the effective range. As discussed in Section 5.1, if we set a higher standard, the effective range becomes narrower, leading to even greater space reduction. An extreme example is when we choose a standard only slightly lower than the optimal performance.