Focusing Where It Matters: Task Reweighting for Generalization in RVT-based Policies

Anonymous CVPR submission

Paper ID *****

Abstract

Robotic manipulation policies often struggle to generalize under real-world distribution shifts. The Colosseum Challenge highlights this gap by evaluating models across diverse scene-level perturbations. While classic models like RVT achieve strong multi-task performance, they still underperform on certain tasks due to uneven training effectiveness. To address this issue, we propose a lightweight task selection and weighting scheme built on RVT, which disables low-transferability tasks and emphasizes generalizable ones. Our method boosts the success rate on clean (variation 0) tasks to 50.4%, outperforming standard baselines by up to 6.8%. Under diverse perturbations, the average success rate increases by 1.3%, demonstrating stronger generalization. Despite a slight increase in the generalization loss, the overall robustness and task-level balance are improved. Our approach requires minimal overhead and can serve as a plug-in to existing RVT-based pipelines.

1. Introduction

Robotic manipulation is a fundamental problem in embodied AI, with wide-ranging applications in industrial automation, service robotics, and assistive technologies. A core challenge in this domain is enabling robots to perform complex, multi-step tasks in diverse, dynamic environments. While recent advances in imitation learning and vision-based control have made significant progress, many models still struggle to generalize beyond their training distributions. To address this, the **Colosseum**[2] benchmark has been introduced as a systematic and large-scale platform to evaluate the generalization capabilities of manipulation policies under a wide range of environmental perturbations. Spanning 20 RLBench tasks and 14 types of scene-level variations—including changes in lighting, texture, object appearance, and physical properties—the benchmark provides a comprehensive setting to evaluate and improve model robustness.

Among recent policy architectures, voxel-based methods such as PerAct[4] and view-based models like RVT[1] have demonstrated strong performance in multi-task manipulation settings. PerAct leverages 3D voxel grids and Perceiver transformers to model spatial features, achieving high accuracy but suffering from high computational costs and limited scalability. In contrast, RVT[1] introduces a multi-view transformer architecture that aggregates re-rendered RGB-D views around the workspace to generate action predictions. RVT strikes a better balance between performance and training efficiency, outperforming PerAct in both training speed and generalization on the RLBench benchmark. However, both approaches tend to underperform on tasks with high variance or low learning signal. In particular, their success rates drop significantly when evaluated under perturbed test distributions such as those defined in the Colosseum benchmark.

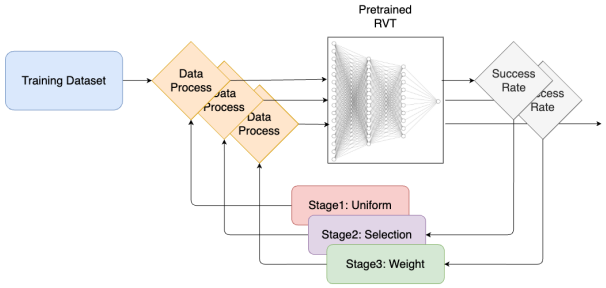

Figure 1. Illustration of the three-stage training procedures: (1) Uniform sampling, (2) Task selection, and (3) Task weighting.

To address these limitations, we propose a lightweight extension to RVT that improves generalization through a task-level selection and weighting mechanism. After training a vanilla RVT model, we analyze task-wise performance and selectively disable tasks with persistently low success rates, while assigning higher weight to those with greater generalization potential. This reweighting effectively shifts the training distribution, encouraging the model to focus on transferable skills rather than overfitting to difficult, low-impact tasks. Our method leads to consistent improvements

in robustness across unseen variations, with minimal additional training cost.

The Colosseum Challenge evaluates generalization in 20 robotic manipulation tasks under unseen environment variations. This report presents our solution, highlighting the following contributions:

1. Reproduced the baseline RVT model on the Colosseum benchmark.
2. Applied post-training with task selection and adaptive weighting strategies.
3. Conducted ablation studies to analyze the impact of our methods.
4. Forked the baseline repository and fixed several bugs, with plans to contribute the improvements as a pull request.

2. Methodology

2.1. Network Architecture

We adopt RVT[1] as the backbone of our policy network. Specifically, we follow the architecture proposed in the original RVT design, which employs a multi-view transformer consisting of an encoder and an action prediction module. The model first reconstructs a 3D point cloud from RGB-D inputs and re-renders the scene from a set of virtual camera viewpoints distributed around the workspace. Each rendered image contains RGB, depth, and 3D coordinate (x, y, z) channels. These images are divided into patches and linearly projected into tokens for transformer-based processing.

The encoder includes eight self-attention layers: the first four apply intra-view attention, allowing tokens to attend only within the same view and extract localized features. The remaining four layers apply global attention across all image tokens, as well as the embedded language tokens that encode the task description, thereby enabling cross-view and language-conditioned reasoning.

For action prediction, the model includes separate output branches. Following RVT, the translation head generates per-view 2D heatmaps from the final image features. These heatmaps are then back-projected into a shared 3D voxel grid, and the end-effector position is selected as the voxel with the highest aggregated score. To predict orientation, Euler angles are discretized into fixed bins, and classification is performed for each rotational axis. The gripper open/close state and a binary collision indicator are predicted using global feature vectors obtained by pooling spatial features across all views, as described in the original RVT framework.

Our implementation follows the original training protocol of RVT, using behavior cloning on expert demonstrations and leveraging data augmentation techniques including 3D rotation and translation of the point cloud prior to rendering. This setup allows for effective learning from a small number of demonstrations while maintaining high inference speed and strong generalization across tasks.

2.2. Technique

We trained our models using behavior cloning (BC). Key techniques include:

- **Task-weighted Sampling:** We dynamically adjusted the training task weights based on model performance, namely the success rate (SR) on different tasks, to focus on more informative tasks. Combined uniform task sampling with Prioritized Experience Replay (PER [3]) to encourage learning from informative samples.
- **Domain Randomization:** Applied random lighting, and camera parameters, which are already included in the training dataset.

2.3. Formal Setup

Let $\mathcal{T}_{\text{all}} = \{\tau_1, \tau_2, \ldots, \tau_N\}$ denote the full task set with N task variations, where $N \leq 20$, and $\mathcal{T}_{\text{excl}} \subset \mathcal{T}_{\text{all}}$ a curated subset that excludes redundant or less informative tasks. For each task τ_i, define its historical success rate $\text{SR}_i \in [0, 1]$ after warm-up post-training phase (11 epochs).

We explore the effect of sampling probability p_i defined as:

$$p_i = \frac{w_i}{\sum_{j \in \mathcal{T}} w_j}, \tag{1}$$

$$\begin{aligned}
\textbf{Uniform:} \quad & w_i = 1 \\
\textbf{Selection:} \quad & w_i = \mathbb{I}\left[\text{SR}_i > 0.1\right] \\
\textbf{Weighting:} \quad & w_i = \mathbb{I}\left(0.1 < \text{SR}_i < 0.9\right) \cdot \left(\mathbb{I}\left[\text{SR}_i^{+}\right] + 1\right)
\end{aligned}$$

where **U**, **S**, and **W** denote three stages of training. For different stages of training, we apply different sampling distribution of training tasks. For details, see section 3.2.

3. Experiments

3.1. Datasets and Evaluation Metrics

We train our model on the official dataset released for the Colosseum Challenge, which contains multi-modal expert demonstrations paired with natural language task descriptions. Except for data augmentation settings, we follow the original RVT implementation for all data-related configurations. Details and rationale for this change are discussed in the next section.

Evaluation is conducted on the official **Colosseum Challenge test set**, which corresponds to the same 20 RLBench tasks used in training. Unlike the training set, each task in the test set includes **14 distinct variations**, introducing scene-level changes such as object positions, distractors, or environmental layouts. These variations are designed to test

the model's ability to generalize beyond the specific conditions seen during training. We adopt the following evaluation pipeline:

1. **Per-Variation Evaluation**: For each variation under a task, we run 25 episodes and compute the success rate as the proportion of successful completions.
2. **Per-Task Averaging**: For each variation type, we compute the average success rate across all 20 tasks, resulting in a per-variation generalization score.
3. **Relative Drop Calculation**: This average is then compared to the model's average success rate on the corresponding standard (non-augmented) task settings. The relative drop in performance is computed as a percentage decrease from the base success rate.
4. **Final Generalization Score (FGS)**: The mean relative performance drop across all 14 variations is reported as the final generalization score, providing a single scalar measure of how well the model transfers to unseen scene configurations.

The FGS highlights the robustness and adaptability of the learned policy in the face of visual and spatial variations, without relying on additional demonstrations or re-training.

3.2. Implementation Details

Our implementation builds upon the official RVT codebase, with minimal modifications to support task-specific weighting and improve training efficiency. The model is implemented using PyTorch, and we employ RLBench and PyRep for simulation and robot-environment interaction. All experiments are conducted on a workstation equipped with two NVIDIA RTX 3090 GPUs.

The model is initially pretrained for 14 epochs, strictly following the original RVT configuration. After the initial training stage, we apply our task selection strategy. Specifically, we set the weights of 6 selected tasks to zero based on prior analysis, while the remaining tasks retain a weight of one. We then train for 11 more epochs using this weighting scheme. We use a batch size of 4 and a learning rate of 5×10^{-5}; training takes about 3 hours.

To further improve training efficiency, we reduce the data augmentation frequency starting from the 14^{th} epoch. This change significantly reduces the memory usage of the replay buffer and shortens the overall training time, without noticeably degrading performance.

In the actual implementation, we further refined the weights of several tasks based on their performance trends during training. For instance, the task *close_box* initially exhibited a declining success rate, which then rebounded as the training progressed. This pattern indicates that there is still substantial potential to enhance the model's capability on this task. Thus, we increased its weight to 2 to emphasize its importance during subsequent training. Conversely, for the *get_ice_from_fridge* task, we observed a concerning trend where the success rate increased initially but then started to decline. This downward trend suggests the onset of overfitting, prompting us to decrease its weight to 1 to reduce the model's overemphasis on this task. Similarly, the *open_drawer* task also showed signs of overfitting with a decreasing success rate after an initial rise, leading us to assign it a weight of 1 as well. The exact task weights used in our implementation are detailed in Table 1.

Table 1. Task Weightings

Task	Weight	Task	Weight
basketball_in_hoop	0	open_drawer	1
close_box	2	slide_block_to_target	0
close_laptop_lid	2	reach_and_drag	1
empty_dishwasher	0	put_money_in_safe	1
get_ice_from_fridge	1	place_wine_at_rack_location	0
hockey	0	insert_onto_square_peg	2
meat_on_grill	0	stack_cups	0
move_hanger	0	turn_oven_on	1
wipe_desk	0	straighten_rope	2
setup_chess	1	scoop_with_spatula	1

4. Ablation Study and Results

In this section, we analyze the effects of **task subset selection** and **sampling weight assignment** on multi-task policy learning and generalization to unseen variations. We conduct an ablation study by training models with and without S (Selection) and W (Task weighting).

4.1. Ablation Variants

We consider the following four settings:

Pattern	Description
–	RVT baseline
UUU	Full task set with uniform sampling ($w_i = 1$)
UUW	Full task set with difficulty-based sampling
USU	Reduced task set with uniform sampling
USW	Reduced task set with difficulty-based sampling

Table 2. Ablation configurations for training policy.

4.2. Results and Analysis

With the USW configuration, which represents our three-stage training approach, the success rate (SR) on variation 0 reaches 50.4%. Compared to ablation baselines, this indicates that the task selection and weighting strategy effectively improves performance on the unperturbed tasks. However, we observe that the average loss (calculated following the provided spreadsheet formulas) also increases slightly with the addition of post-training and task reweighting strategies.

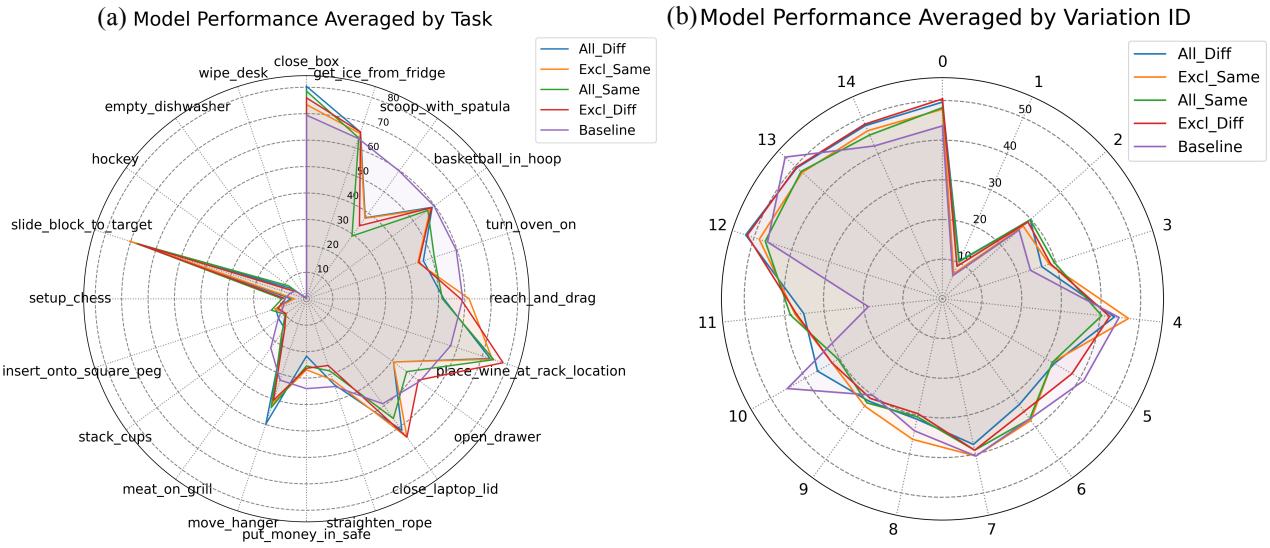

Figure 2. Success Rate Comparison of Baseline and Models with techniques. Excl/All represent using/not using the task selection. Diff/Same stands for the weight of tasks in the third stage.

To better evaluate generalization under realistic settings, we compute the average SR across all types of perturbations. The results, summarized in Tab. 3, show that our techniques yield an improvement of approximately 1.3% in average SR under variations, demonstrating their practical benefit. Comprehensive results for each setting are provided in Tables 4, 5, 6, and 7.

The SR of these five variants across different perturbations is depicted in Fig. 2. We found that the tasks assigned a weight of 2 during the third stage perform significantly better comparing to the baseline in general.

Table 3. Evaluation of models

Method	SR (%)		Avg. Loss
	variation0	1-14	
Baseline	43.6	34.3	-0.2011
UUU	48.2	34.9	-0.2553
USU	47.8	35.6	-0.2238
UUW	49.6	35.6	-0.2588
USW	50.4	35.4	-0.2742

The combination of a curated task subset $\mathcal{T}_{\text{excl}}$ with difficulty-aware reweighting emerges as a practical middle ground for scalable, efficient, and generalizable multi-task policy learning.

5. Conclusion

Robotic manipulation models, such as RVT, often struggle to generalize under real-world distribution shifts. To address this issue, we proposed a lightweight task selection and weighting scheme built upon RVT. Following the initial training phase, we deactivated tasks exhibiting low transferability and prioritized those with higher generalization potential, thereby reorienting the training distribution.

Evaluated on the Colosseum benchmark, our method significantly improved model robustness, achieving higher performance under perturbations at minimal additional cost. This work highlights the importance of task-level analysis in training and offers a practical solution for enhancing generalization. In future research, we plan to integrate our task weighting mechanism with other techniques to further advance robotic generalization capabilities.

References

[1] Ankit Goyal, Jie Xu, Yijie Guo, Valts Blukis, Yu-Wei Chao, and Dieter Fox. Rvt: Robotic view transformer for 3d object manipulation. In *Conf. Robot Learning*, pages 694–710. PMLR, 2023. 1, 2

[2] Wilbert Pumacay, Ishika Singh, Jiafei Duan, Ranjay Krishna, Jesse Thomason, and Dieter Fox. The colosseum: A benchmark for evaluating generalization for robotic manipulation. *arXiv preprint arXiv:2402.08191*, 2024. 1

[3] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay, 2016. 2

[4] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023. 1

Task Name	No variations	All variations	MO_Color	RO_Color	MO_Texture	RO_Texture	MO_Size	RO_Size	Light-color	Table-Color	Table-texture	Distractor	Backgrond-texture	RLBench variations	Camera pose
basketball_in_hoop	100	0	100	4	76	-	92	76	40	28	32	16	96	80	80
close_box	76	52	28	-	-	-	92	-	100	84	92	68	80	76	88
close_laptop_lid	64	16	68	-	-	-	40	-	76	84	88	64	72	76	64
empty_dishwasher	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0
get_ice_from_fridge	80	0	84	16	84	-	72	64	64	64	56	76	92	88	88
hockey	0	8	0	0	-	0	0	0	8	32	0	8	0	0	4
meat_on_grill	4	64	8	40	-	-	4	-	8	20	4	4	4	4	4
move_hanger	100	0	0	100	-	-	-	-	0	0	4	40	100	100	0
wipe_desk	0	0	0	-	0	-	0	-	0	0	0	0	0	0	0
open_drawer	84	4	4	-	-	-	56	-	48	24	48	80	84	64	80
slide_block_to_target	96	24	36	-	80	-	-	-	48	44	36	72	100	100	100
reach_and_drag	80	0	32	40	80	92	76	36	12	20	60	64	96	92	96
put_money_in_safe	20	0	64	0	12	8	8	-	16	20	32	44	44	36	64
place_wine_at_rack_location	92	12	92	48	-	92	48	80	96	88	96	72	88	92	100
insert_onto_square_peg	32	0	0	20	-	20	28	4	4	4	4	12	4	16	12
stack_cups	8	0	4	-	12	-	4	-	16	28	12	8	4	12	8
turn_oven_on	48	0	48	-	-	-	28	-	32	32	44	68	48	68	76
straighten_rope	52	0	8	-	36	-	-	-	8	12	4	32	60	44	36
setup_chess	16	0	0	16	4	-	16	-	0	4	12	8	12	8	8
scoop_with_spatula	56	0	4	60	40	52	32	52	16	32	20	4	52	36	56

Table 4. Results for RVT-USW for various perturbations

Task Name	No variations	All variations	MO_Color	RO_Color	MO_Texture	RO_Texture	MO_Size	RO_Size	Light-color	Table-Color	Table-texture	Distractor	Backgrond-texture	RLBench variations	Camera pose
basketball_in_hoop	96	8	100	4	76	-	92	76	36	24	32	4	96	80	72
close_box	84	56	28	-	-	-	88	-	96	92	92	76	84	92	76
close_laptop_lid	64	16	68	-	-	-	36	-	64	80	56	64	40	52	76
empty_dishwasher	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0
get_ice_from_fridge	72	0	84	20	84	-	80	52	68	68	68	72	72	84	72
hockey	8	16	8	4	-	0	8	0	4	48	4	8	0	0	8
meat_on_grill	4	68	20	44	-	-	4	-	4	20	8	4	4	4	4
move_hanger	100	0	0	100	-	-	-	-	0	0	4	68	100	100	4
wipe_desk	0	0	0	-	0	-	0	-	0	0	0	0	0	0	0
open_drawer	52	0	0	-	-	-	32	-	60	36	56	76	80	64	60
slide_block_to_target	96	24	24	-	80	-	-	-	60	44	40	80	96	100	100
reach_and_drag	80	0	20	36	68	60	80	44	8	8	56	64	80	96	72
put_money_in_safe	28	0	60	0	8	20	12	-	16	12	24	44	32	28	72
place_wine_at_rack_location	92	8	84	56	-	80	48	92	80	92	76	72	76	92	96
insert_onto_square_peg	24	0	0	28	-	24	48	12	0	0	4	4	20	16	16
stack_cups	12	0	8	-	4	-	0	-	16	32	4	8	16	8	12
turn_oven_on	60	4	44	-	-	-	48	-	48	52	44	72	52	68	68
straighten_rope	44	4	44	-	40	-	-	-	8	8	4	40	60	32	32
setup_chess	8	0	0	8	16	-	20	-	4	0	8	12	12	16	16
scoop_with_spatula	40	0	4	56	28	40	40	36	36	32	28	4	20	28	48

Table 5. Results for RVT-UUU for various perturbations

Task Name	No variations	All variations	MO_Color	RO_Color	MO_Texture	RO_Texture	MO_Size	RO_Size	Light-color	Table-Color	Table-texture	Distractor	Backgrond-texture	RLBench variations	Camera pose
basketball_in_hoop	100	4	100	4	72	-	100	76	44	32	32	12	96	80	72
close_box	88	64	28	-	-	-	88	-	96	88	92	76	88	84	92
close_laptop_lid	64	24	72	-	-	-	24	-	68	80	80	68	68	68	60
empty_dishwasher	0	4	4	0	-	0	0	0	0	0	0	0	0	0	0
get_ice_from_fridge	84	0	80	28	88	-	80	52	60	68	64	80	84	84	76
hockey	0	16	4	0	-	0	4	0	12	48	0	4	0	0	0
meat_on_grill	4	48	16	36	-	-	4	-	16	20	8	4	4	4	4
move_hanger	100	0	0	100	-	-	-	-	0	0	100	44	100	100	4
wipe_desk	0	0	0	-	0	-	0	-	0	0	0	0	0	0	0
open_drawer	64	0	0	-	-	-	28	-	48	20	36	56	84	48	64
slide_block_to_target	96	40	52	-	68	-	-	-	60	44	40	72	100	100	100
reach_and_drag	68	0	28	32	72	68	60	48	20	12	52	64	84	88	84
put_money_in_safe	24	0	56	0	12	4	0	-	4	24	20	32	32	32	64
place_wine_at_rack_location	84	8	76	44	-	92	56	72	84	92	88	64	80	88	96
insert_onto_square_peg	16	0	0	12	-	16	48	8	0	8	0	4	20	24	16
stack_cups	20	0	12	-	16	-	0	-	28	8	8	8	16	24	16
turn_oven_on	60	0	36	-	-	-	28	-	36	56	40	64	56	72	64
straighten_rope	56	0	24	-	52	-	-	-	8	4	16	40	64	52	60
setup_chess	8	0	0	8	8	-	8	-	4	4	12	0	20	0	4
scoop_with_spatula	56	0	8	52	48	48	32	44	32	28	40	12	48	40	80

Table 6. Results for RVT-UUW for various perturbations

Task Name	No variations	All variations	MO_Color	RO_Color	MO_Texture	RO_Texture	MO_Size	RO_Size	Light-color	Table-Color	Table-texture	Distractor	Backgrond-texture	RLBench variations	Camera pose
basketball_in_hoop	92	4	100	4	76	-	80	80	48	32	40	16	92	72	76
close_box	84	24	12	-	-	-	92	-	96	96	88	80	80	80	76
close_laptop_lid	64	12	72	-	-	-	28	-	80	80	96	64	64	76	76
empty_dishwasher	0	4	0	0	-	0	0	0	0	0	0	0	0	0	0
get_ice_from_fridge	72	0	80	20	88	-	72	64	88	68	64	84	76	80	64
hockey	0	8	0	0	-	0	0	0	12	32	4	4	0	0	0
meat_on_grill	4	60	16	44	-	-	4	-	24	20	4	4	4	4	4
move_hanger	100	0	0	100	-	-	-	-	0	0	4	56	100	100	0
wipe_desk	0	0	0	-	0	-	0	-	0	0	0	0	0	0	0
open_drawer	56	0	0	-	-	-	52	-	36	12	16	64	80	64	68
slide_block_to_target	96	16	40	-	88	-	-	-	64	52	44	76	100	100	100
reach_and_drag	80	0	28	36	84	72	84	44	60	32	76	84	84	76	84
put_money_in_safe	32	0	64	0	16	4	8	-	20	12	20	40	36	36	88
place_wine_at_rack_location	96	8	68	56	-	80	48	92	76	96	84	76	84	84	88
insert_onto_square_peg	8	0	0	16	-	12	60	8	8	12	0	4	16	16	16
stack_cups	8	4	16	-	12	-	0	-	16	8	8	4	20	0	16
turn_oven_on	64	0	20	-	-	-	36	-	36	52	56	60	40	64	60
straighten_rope	56	0	28	-	52	-	-	-	12	20	8	24	52	48	56
setup_chess	4	0	0	8	12	-	8	-	0	0	4	4	8	8	4
scoop_with_spatula	40	0	4	56	44	56	72	36	48	44	28	4	36	44	52

Table 7. Results for RVT-USU for various perturbations