

864	<b>Appendix</b>	
865		
866	<b>A Overview video</b>	18
867		
868	<b>B Additional qualitative results</b>	18
869		
870		
871	<b>C Additional related work.</b>	18
872		
873	<b>D Dataset Details.</b>	18
874		
875	<b>E Additional downstream evaluations: Zero-shot cross-modal retrieval task</b>	19
876		
877	<b>F TST-MM deployment in the wild.</b>	20
878		
879		
880	<b>G TST with the DINOv2 objective.</b>	21
881		
882	<b>H Benchmarking different self-supervised objectives under TST.</b>	21
883		
884	<b>I Using multimodal data during transfer: are additional modalities all you need?</b>	21
885		
886		
887	<b>J What is the smallest unit of space we can specialize on?</b>	22
888		
889	<b>K TST with synthetic data.</b>	23
890		
891	<b>L The role of the transfer dataset mix-in during pre-training.</b>	24
892		
893	<b>M TST with off-the-shelf transfer set.</b>	24
894		
895		
896	<b>N Pseudo-labeler baselines</b>	25
897		
898	<b>O Additional baselines</b>	25
899		
900	<b>P Unimodal specialization-generalization</b>	26
901		
902	<b>Q Do other self-supervised objectives benefit from specialized pre-training?</b>	26
903		
904		
905	<b>R Is distilling in the test space beneficial?</b>	27
906		
907	<b>S Application for hardware data collection</b>	27
908		
909	<b>T Test time training with TST</b>	28
910		
911	<b>U Continual learning with TST.</b>	28
912		
913		
914	<b>V Sampling ratio between test space and transfer data during TST pre-training</b>	29
915		
916	<b>W Experimental Setup details</b>	30
917	<b>W.1 Pre-training details</b> . . . . .	30

918	<b>W.2 Transfer details</b> . . . . .	31
-----	---------------------------------------	----

919		
920	<b>X Computational Resources.</b>	32

## 921 A OVERVIEW VIDEO

922 We provide a video with narration that gives a high-level summary of our paper. **We recommend watching the video.** The video can be found in the supplementary zip folder.

## 923 B ADDITIONAL QUALITATIVE RESULTS

924 We provide additional qualitative results in Fig 24 and Fig 25. We also provide more video results on various tasks in the supplementary zip file.

## 925 C ADDITIONAL RELATED WORK.

926 **Domain Adaptation** in vision (Li et al., 2017; Zhou et al., 2021) addresses the gap between a source domain, where abundant data is available, and the target domain, where limited (Shu et al., 2019; Liu et al., 2024) or no data (Dong et al., 2021; Ganin & Lempitsky, 2014) are available. TST, when initialized from an Internet-based model, as presented in Tab. 2, can be seen as an instantiation of adapting a generalist model to the test space. However, TST differs by learning task-agnostic representations by self-supervised pre-training in the test space, as opposed to domain adaptation, which generally adapts a pre-trained task-specific network (Xu et al., 2021; Kang et al., 2019).

927 **Semi-Supervised Learning** refers to a line of work that attempts to learn a task from a limited labeled dataset and massive unlabeled data (van Engelen & Hoos, 2019). Clearly, it involves consistency regularization (Berthelot et al., 2019; Sohn et al., 2020; Xie et al., 2019) and pseudo-labeling (Guo et al., 2022; Chen et al., 2021; Zhang et al., 2021; Liu et al., 2021a) to generate supervision of unlabeled data, followed by joint training. Our framework, TST is closer in spirit to *self-supervised learning*, as it tries to learn a task-agnostic representation for the test space, that we transfer for various downstream tasks like segmentation, detection and image captioning. Under semi-supervised learning, specialization with TST can be posed as using unlabelled data from the test space, as opposed to other sources like Internet or similar spaces.

928 **Embodied Active learning.** In another line of work, SEAL (Chaplot et al., 2021), Interactron (Kotar & Mottaghi, 2022) learn a reinforcement learning based policy to collect supervision in a house to finetune an off-the-shelf MaskRCNN (He et al., 2017), or observe additional frames for multi-frame inference for object detection. As opposed to focusing on adapting task-specific models, we focus on learning task-agnostic pre-trained representations over a test space.

## 929 D DATASET DETAILS.

930 **1. Scannet++** (Yeshwanth et al., 2023) is a large dataset of real-world indoor spaces containing sub-millimeter resolution laser scans, paired with DSLR and iPhone RGB images.

- 931 • **Pre-training dataset.** We use 8 Scannet++ (Yeshwanth et al., 2023) scenes as our test space. We use a mix of iPhone and DSLR images for pre-training, with the iPhone containing 19165 samples and the DSLR dataset containing 15000 samples.
- 932 • **Transfer dataset.** We use non-test space buildings for creating a transfer set of 40000 RGB, segmentation pairs. Note that Scannet++ (Yeshwanth et al., 2023) only provides 3D instance annotations, which we project to 2D to create a semantic segmentation dataset.
- 933 • **Evaluation.** We evaluate on semantic segmentation in the test space. The test dataset for evaluation contains 3000 RGB image samples. Note that we collect a separate held out set from the test space for this stage.

934 **2. Replica** (Straub et al., 2019) provides high quality 3D reconstructions of real indoor spaces.

- **Pre-training dataset.** We use Omnidata (Eftekhar et al., 2021), to densely sample Replica meshes corresponding to the 5 scenes to build our pre-training dataset,  $D_{PT}$ , containing 84889 samples. We defer the details of the sampling procedure to Omnidata (Eftekhar et al., 2021).
- **Transfer dataset.** Similar to Scannet++ (Yeshwanth et al., 2023), we collect a transfer set from another set of Replica scenes that are different than the scenes used during pre-training. We collect 20000 RGB images and semantic segmentation masks, and use it as our transfer dataset,  $D_t$ .
- **Evaluation.** We evaluate on semantic segmentation in the test space. We collect a test set of 5000 images and semantic segmentation annotations from the same test space we pre-train on, and report performance on it. We leverage Omnidata annotation pipeline to extract the segmentation labels.

**3. ProcTHOR** (Deitke et al., 2022) It includes procedurally generated house-like environments. We use 5 procedurally generated houses as our test space.

- **Pre-training dataset.** We randomly sample various agent  $x, y, z$  positions and orientations along its axis in the test space, and collect RGB-D images at these points. This sampling process yields a total of 163767 samples. We collect data by sampling densely across the test space and use it as our pre-training dataset  $D_{PT}$ .
- **Transfer dataset.** For the transfer data  $D_t$ , we collect a small dataset of 20000 RGB and task annotation pairs, from 800 houses generation using a different asset and layout distribution than the pre-training test space, thereby making them out-of-distribution to it.
- **Evaluation.** We evaluate TST and present results on three tasks, namely semantic segmentation, object detection and image captioning. We collect a test set with 5000 samples from the same test space, where we performed pre-training, and report performance on it. We use the AI2-THOR (Kolve et al., 2017) metadata to extract semantic segmentation and object detection labels for evaluations. For captioning, we generate ground truth captions by prompting GPT-4o (OpenAI, 2024) with privileged information, e.g. class names and bounding boxes. Finally, we additionally evaluate our model on cross-modal retrieval (in Sec. E).

## E ADDITIONAL DOWNSTREAM EVALUATIONS: ZERO-SHOT CROSS-MODAL RETRIEVAL TASK

Method	Image to Depth			Depth to Image		
	R@1	R@5	R@10	R@1	R@5	R@10
4M-21 (Bachmann et al., 2024)	1.06	2.18	3.08	1.0	2.76	3.66
TST-MM	<b>25.48</b>	<b>37.00</b>	<b>41.58</b>	<b>24.32</b>	<b>36.46</b>	<b>40.82</b>

Table 3: **Zero-shot Cross-modal retrieval.** When performing the image-to-depth and depth-to-image cross-modal retrievals on the test space data using the predicted CLIP embeddings, we observe that the TST-MM method constantly outperforms the Internet-based 4M-21 (Bachmann et al., 2024).

As mentioned in Section 4.1, we present results on zero-shot cross-modal retrieval to further support our framework TST. Specifically, we evaluate the performance of models pre-trained with TST-MM on RGB-to-Depth and Depth-to-RGB retrieval. To perform retrieval using an Internet-based model, 4M-21 (Bachmann et al., 2024) and TST-MM, we utilize their cross-modal generation capabilities by transforming depth and RGB images into CLIP embeddings, and then apply retrieval directly on the CLIP embeddings. Since 4M-21 (Bachmann et al., 2024) and TST-MM generate feature maps for CLIP as the target modality from RGB and Depth images, we apply mean-pooling on the feature maps to obtain global CLIP embeddings. Cross-Modal retrieval evaluates TST-MM on two fronts: i) How well test-space paired modality inputs are aligned in the model representations internally, and ii) How effectively TST-MM can perform cross-modal generalization. For the evaluation, we report zero-shot recall at various thresholds on a test set of 5000 samples from ProcTHOR (Deitke et al.,



Figure 10: **TST-MM cross-modal retrieval predictions.** TST-MM retrieves corresponding RGB images from query Depth input and Depth images from RGB input more accurately than the Internet based 4M-21 (Bachmann et al., 2024) model.

(2022) test space. The results are presented in Tab. 3. We also present qualitative examples in Fig. 10. Note that given our method TST-MM has access to the test space, it can retrieve RGB to Depth and Depth to RGB much more effectively than models based on external data like the Internet.

We find that TST-MM substantially outperforms 4M-21 (Bachmann et al., 2024). The recall performance of TST-MM further increases when evaluated on R@5 and R@10, whereas Internet-based 4M-21 (Bachmann et al., 2024) shows diminishing returns. This underscores the effectiveness of test-space training, where specialization itself is crucial for learning test-space-aligned representations.

## F TST-MM DEPLOYMENT IN THE WILD.

In addition to real-world results on Scannet++ 4.5, we also experiment with the deployment of TST, in a custom space. We collect a 15-minute video of a meeting room and used the resulting frames for pre-training described in Sec. 4.1 followed by a transfer on the ScanNet++ (Yeshwanth et al., 2023) transfer set (Sec. D). We evaluated TST-MM and the baselines on the semantic segmentation task. We evaluate TST-MM and the baselines on the semantic segmentation task. Tab. 4 shows that for this custom scene deployment, pre-training on the test-space through TST-MM outperforms the Internet-based baseline 4M-21 (Bachmann et al., 2024). The qualitative comparison in Fig. 11 shows that TST-MM’s predictions are notably better than those of the Internet-based 4M-21 (Bachmann et al., 2024).

Method	mIoU
Scratch	21.82
4M-21	54.58
TST-MM	<b>59.11</b>

Table 4: **Semantic segmentation performance.** Comparison of mIoU scores across different training methods.

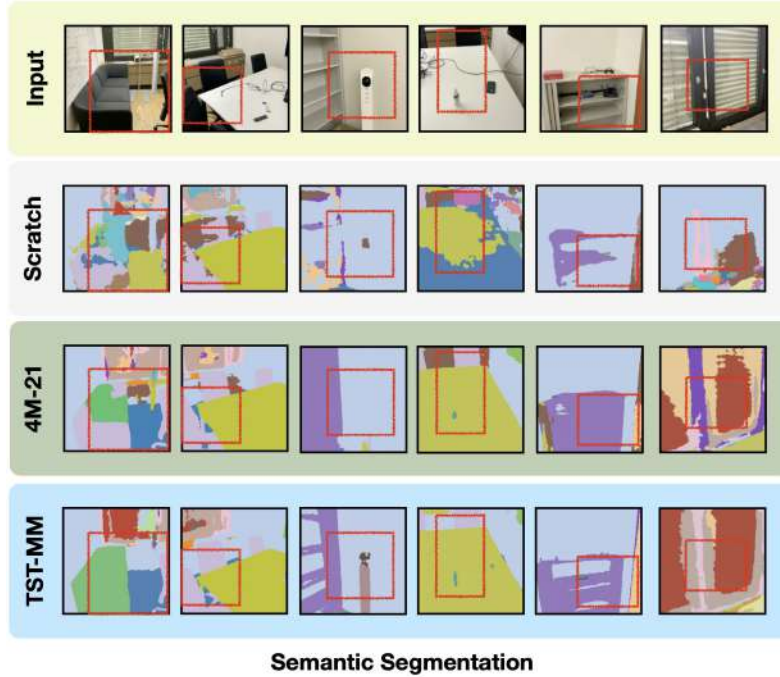


Figure 11: **TST-MM predictions on deployment in the wild.** We showcase the qualitative results for TST-MM on the semantic segmentation task against the Internet-based pre-trained model 4M-21 (Bachmann et al., 2024) and scratch (no-pretraining). TST-MM predictions are notably better across object categories, showing the value of access to test space and the deployment potential of TST-MM.

## G TST WITH THE DINOv2 OBJECTIVE.

In this section, we explore how TST trained with DINOv2 objective, TST-DINO from *scratch*, compares with its Internet counterpart trained on 142M images from the Internet (Oquab et al., 2023). Fig. 16 shows that pre-training on only data from the test space can substitute large-scale Internet pre-training. This further underscores that TST framework extends to other self-supervised objectives (Oquab et al., 2023) beyond masked modeling for specialization. However, we find that TST-MM, which uses multimodal masked modeling outperforms other unimodal self-supervised objectives like DINOv2 (Oquab et al., 2023) and MAE (He et al., 2021).

## H BENCHMARKING DIFFERENT SELF-SUPERVISED OBJECTIVES UNDER TST.

We compare the performance of TST with different pre-training objectives such as multimodal masked modeling (Bachmann et al., 2024), DINOv2 (Oquab et al., 2023) and MAE (He et al., 2021). As shown in Fig. 17, we find that multimodal masked modeling (TST-MM) to be the most performant among the self-supervised objectives we explored. However, note that all 3 objectives show specialization trends as presented in Fig. 8 and Fig. 19.

## I USING MULTIMODAL DATA DURING TRANSFER: ARE ADDITIONAL MODALITIES ALL YOU NEED?

In Sec. 4.5 we discuss that pre-training on multimodal data with multimodal masked modeling objective in our TST-MM method leads to a specialist model more performant than other baselines. Here we check if this superior performance is solely due to access to the additional modalities besides RGB that simplify the task, rather than representation learning value through multimodal *pre-training*?

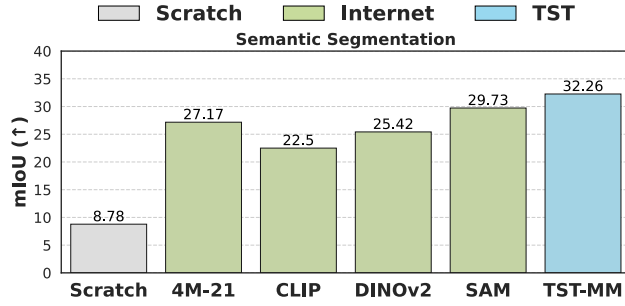


Figure 12: **TST works with off-the-shelf transfer set.** For Replica (Straub et al., 2019), we find that even when we use ADE20k (et al., 2017) as a transfer set, TST-MM outperforms Internet-based generalist models, showcasing the importance of having access to the test space, agnostic to the transfer set.

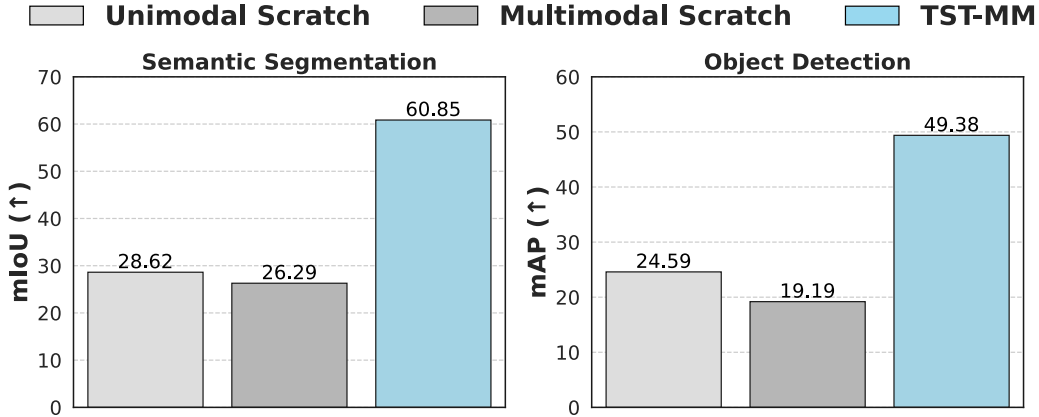


Figure 13: **Multimodal pre-training is crucial in TST-MM.** We compare our method to the model that also has access to multimodal data during supervised training on transfer data. “Multimodal Scratch” uses all modalities, including RGB, as input and predicts the corresponding semantic segmentation map during both training and testing. TST-MM, which uses only RGB as input during transfer, significantly outperforms the multimodal scratch model, signifying the importance of multimodal pre-training.

Figure 13 shows the performance of a model trained from scratch, only on the semantic segmentation task using multimodal data, ie, no self-supervised pre-training. It receives all of the modalities as input and predicts the corresponding segmentation map during both training and testing. We find that this model performs poorly compared to TST-MM, which leverages multimodal data during pre-training and only RGB input during transfer and test. This experiment signifies the importance of pre-training using multimodal data from the test space.

## J WHAT IS THE SMALLEST UNIT OF SPACE WE CAN SPECIALIZE ON?

In the results presented so far, we have shown that TST can specialize on test spaces at the size from 1-8 houses. However, this raises a question, what is the smallest unit of space we can specialize on? To probe this, we reduce the size of the test space and evaluate if TST can specialize to it. We consider a model trained via TST specialized, if it can outperform an off-the-shelf Internet-based generalist, when evaluated on that test space. We reduce the test space, in the form of concentric rectangles, starting with a room, and then reducing the size of the rectangle. For each rectangle, we pre-train a specialist model via TST. We compare this against 4M-21 (Bachmann et al., 2024), on the task of semantic segmentation. We find that we can specialize on a single room (ring 3) that has an



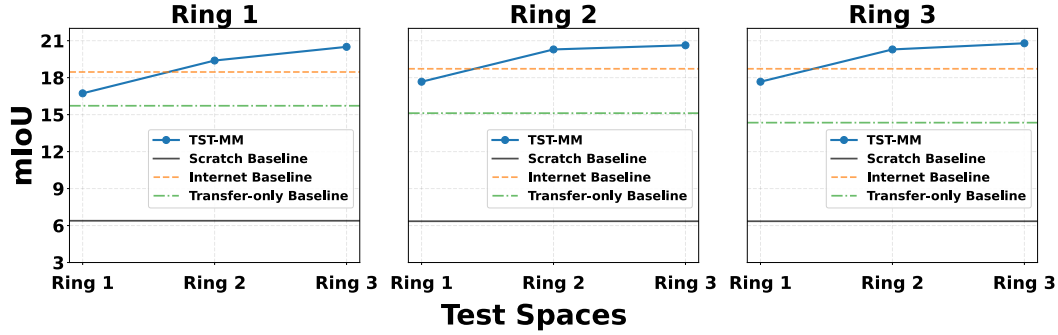


Figure 14: **Smallest unit of space to specialize on.** We reduce the test space size, that we can specialize and pre-train models with TST-MM. We compare it with an Internet pre-trained model (Bachmann et al., 2024), and a baseline that pre-trains only on the transfer set. We also find that training on a ring smaller than the test ring, leads to diminished performance.

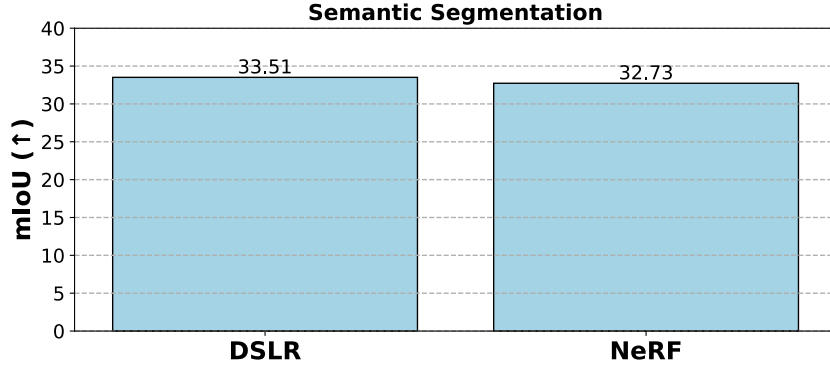


Figure 15: **TST with synthetic data.** We replace real DSLR images in ScanNet++ (Yeshwanth et al., 2023) with NeRF (Mildenhall et al., 2020)-rendered images from the same training viewpoints. We find that this results in only negligible performance, hence demonstrating that NeRF’s output quality at known poses is sufficient to substitute high-quality DSLR images.

area of 20 square metres, and this trend continues as we reduce it down to ring 2, which is 11 square metres and ring 1 which is 5 square metres. Reducing the test space, below 5 square metres results in failed specialization, where the pre-training on just the transfer pre-training performs the best.

## K TST WITH SYNTHETIC DATA.

Recent advances in novel view synthesis (Mildenhall et al., 2020; Barron et al., 2021; Kerbl et al., 2023) have enabled realistic renderings of indoor spaces, opening up the potential for generating synthetic training data. In TST, we leverage existing indoor scene datasets (Yeshwanth et al., 2023; Straub et al., 2019), which include real RGB images captured with DSLR/iPhone cameras or rendered from 3D meshes, to develop specialized models for specific test spaces. This leads to a key question: if a novel view synthesis model can generate images from arbitrary viewpoints in a test space, can it serve as a controllable data generator—and can its outputs match real images in utility?

To explore this, we train a NeRF model (Tancik et al., 2023) using DSLR images from ScanNet++ (Yeshwanth et al., 2023), and render images from the same camera poses. We then pre-train two models—one using real DSLR images and the other using NeRF-rendered views—to assess the performance gap. As shown in Fig. 15, NeRF-generated images result in negligible performance loss compared to real images. This suggests an interesting future direction: if high-fidelity NeRF models can be trained with fewer input images, they could act as steerable data generators, reducing the need for extensive real-world data collection in test environments.

Table 5: **Ablating the use of transfer RGB frames during pre-training.** As noted in Sec. 4.1 we additionally use RGB images from the transfer set during pre-training. We ablate this choice by comparing all three dataset configurations. We use the ViT-S backbone for all models.

	Test Space	Transfer	Segmentation (mIoU $\uparrow$ )
	$\times$	$\checkmark$	42.01
TST	$\checkmark$	$\times$	50.21
	$\checkmark$	$\checkmark$	<b>56.96</b>
4M-21			46.12

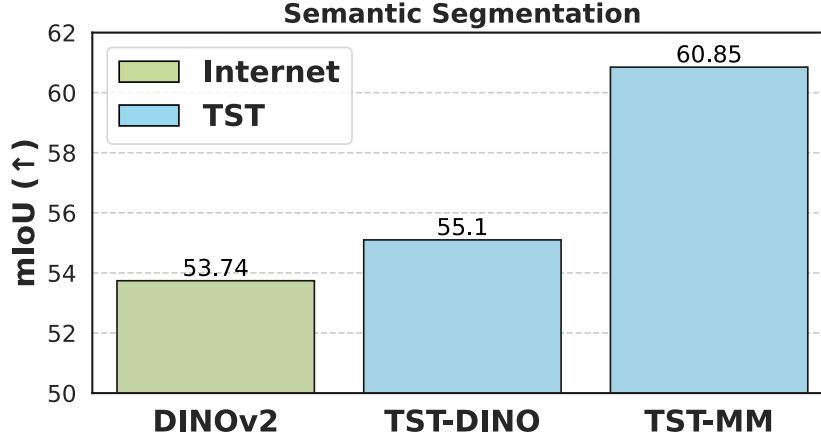


Figure 16: **TST with DINOv2 objective outperforms its Internet counterpart.** We compare the performance of DINOv2 pre-training in the test space, TST-DINO, with DINOv2 pre-trained on the large-scale Internet dataset of 142M images (Oquab et al., 2023). TST-DINO outperforms its Internet counterpart, showing the value of specialization. Yet, TST-MM with multimodal masked modeling achieves the best performance.

## L THE ROLE OF THE TRANSFER DATASET MIX-IN DURING PRE-TRAINING.

We study the role of mixing images from the test space and transfer datasets during pre-training, as mentioned in Sec. 4.1. Tab. 5 shows that using only test-space data outperforms both pre-training on large-scale Internet data and using only transfer images, but mixing test space and transfer data achieves the best performance. We hypothesize that seeing transfer images during pre-training helps the model to better align with the fine-tuning stage on the transfer dataset. Note that it cannot be explained by more diverse data in the transfer set, as adding non-test spaces decreases the specialization performance, as observed in Fig. 7.

## M TST WITH OFF-THE-SHELF TRANSFER SET.

As noted in Sec. 4.1, for each dataset (Deitke et al., 2022; Straub et al., 2019; Yeshwanth et al., 2023) we explore, the transfer set comes from a similar domain, as the pre-training dataset, albeit from non-test spaces. It naturally raises the question, what if we use an existing off-the-shelf semantic segmentation dataset like ADE20k (et al., 2017) as a transfer set. Does TST generalize and result in performant specialist models, or is an in-domain transfer set necessary? To probe this, for the Replica (Straub et al., 2019) dataset, we pre-train TST-MM, but instead of using non-test spaces from Replica as the transfer set, we use ADE20k (et al., 2017). Fig. 12 shows TST-MM outperforms various generalist models (Bachmann et al., 2024; Oquab et al., 2023; Radford et al., 2021), even when using ADE20k (et al., 2017) as the transfer set. All models are evaluated in the test space from Replica (Straub et al., 2019), on semantic segmentation, with a ViT-B backbone.



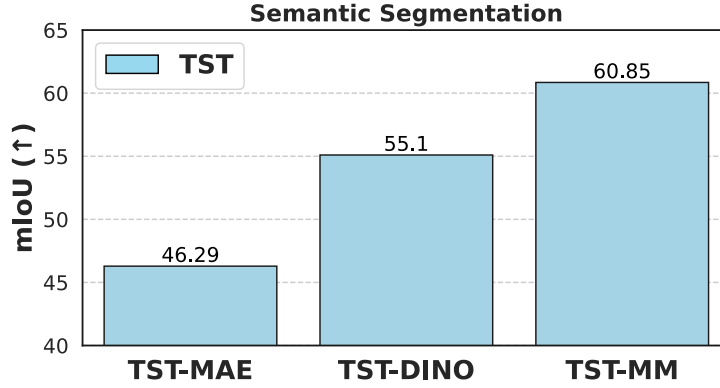


Figure 17: Comparison between different pre-training objectives under the TST framework. We compare the performance of different pre-training objectives using TST on the semantic segmentation task. We find that multimodal masked modeling (TST-MM) achieves the best performance followed by TST-DINO. All the three objectives were trained using the ViT-B model size on the ProcTHOR (Deitke et al., 2022) dataset.

Method		Semantic Segmentation			Object Detection		Captioning	
		Scannet++ mIoU ↑	ProcTHOR mIoU ↑	Replica mIoU ↑	Scannet++ mAP ↑	ProcTHOR mAP ↑	ProcTHOR CIDEr ↑	SPICE ↑
Pseudo-labelers	ImageBind (Girdhar et al., 2023)	25.40	44.54	12.78	6.78	32.54	-	-
	CLIP (Radford et al., 2021)	23.02	48.66	20.92	19.75	38.47	18.4	16.2
	Mask2Former (Cheng et al., 2022)	29.42	50.28	22.68	-	-	-	-
	ViTDet (Li et al., 2022)	-	-	-	23.49	44.10	-	-
	SAM (Kirillov et al., 2023)	34.75	56.72	28.51	-	-	-	-
Specialist	TST-MM	34.49	<b>60.85</b>	32.87	31.54	49.38	34.3	20.4
Pre-training	TST-MM (adapted)	<b>36.44</b>	<b>60.59</b>	<b>34.53</b>	<b>35.83</b>	<b>51.25</b>	<b>39.9</b>	<b>20.5</b>

Table 6: Comparing TST-MM against pseudolabels. We find that TST-MM outperforms all pseudolabels underscoring the value of pre-training on them via multimodal masked modelling in the test space.

## N PSEUDO-LABELER BASELINES

As mentioned in Sec. 4.5, we use various off-the-shelf networks to pseudolabel RGB data, and create additional (optional) modalities for TST-MM. We present a comparison for TST-MM against these pseudolabel baselines in Tab. 6. TST-MM and TST-MM (adapted) outperform all pseudolabel baselines, suggesting the benefit of pre-training in the test space with them, via multimodal masked modeling.

## O ADDITIONAL BASELINES

TST-MM includes modalities obtained as outputs from different off-the-shelf models. Tab. 2 shows that TST-MM outperforms each individual model used as a modality. Since our transfer tasks are semantic segmentation and object detection, we further study if having off-the-shelf models trained on related tasks as modalities is crucial for our final performance.

We present three experiments using the ViT-B backbone on ProcTHOR (Deitke et al., 2022). For each experiment, we drop one of the following modalities: i) Semantic segmentation, ii) Object detection, iii) Semantic segmentation, Object detection, and SAM edges. Tab. 7 shows the results for each model when transferred to semantic segmentation and object detection. We find that even though the performance drops if we remove all three modalities, TST-MM still outperforms the Internet-based 4M-21 (Bachmann et al., 2024) model.

Method	Modalities				Task	
	Semantic segmentation	Object detection	SAM edge	Others	Segmentation (mIoU $\uparrow$ )	Detection (mAP $\uparrow$ )
TST-MM	✓	✓	✓	✓	<b>60.85</b>	49.38
	✗	✓	✓	✓	59.43	<b>49.58</b>
	✓	✗	✓	✓	59.38	49.34
	✗	✗	✗	✓	55.39	45.97
4M-21 (Bachmann et al., 2024)	✓	✓	✓	✓	53.24	41.43

Table 7: **The effect of semantic modalities in TST-MM.** As the results demonstrate, removing the semantic segmentation and object detection modalities obtained from off-the-shelf networks does not significantly hurt the TST-MM’s performance on the downstream semantic segmentation and object detection tasks. When all three semantic modalities are removed, we observe a drop in performance, but TST-MM still outperforms the Internet-based 4M-21 (Bachmann et al., 2024) model, demonstrating the value of specialization.

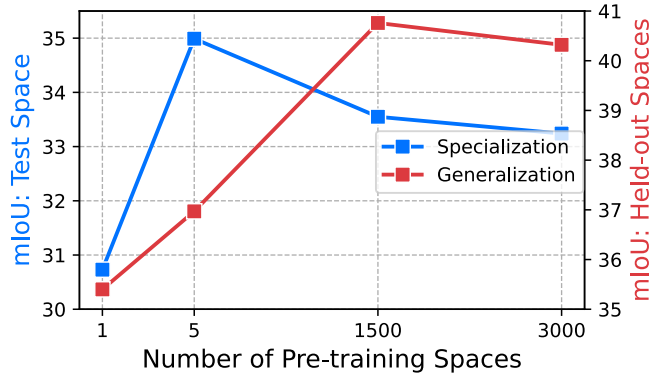


Figure 18: **Unimodal specialization vs generalization.** We further show the specialization-generalization trend with unimodal pre-training.

## P UNIMODAL SPECIALIZATION-GENERALIZATION

In Sec. 4.4, we presented the results for specialization-generalization trade-off via multimodal pre-training (Fig. 7). In this section we further examine the specialization-generalization trend under unimodal pre-training, where we pre-train using RGB as the only modality.

The results are presented in Fig. 18 demonstrate that in the unimodal pre-training regime there’s an opposite specialization trend compared to the multimodal pre-training shown in Fig. 7. This further shows the importance of multimodality in order to achieve a performant model in case of specialization.

## Q DO OTHER SELF-SUPERVISED OBJECTIVES BENEFIT FROM SPECIALIZED PRE-TRAINING?

In Sec. 4.5, we present results with TST-MM, which employs multimodal masked modeling. However, as mentioned in Sec. 3.3, TST also supports other self-supervised objectives. Fig. 19 shows that pre-training objectives, DINOv2 (Oquab et al., 2023), and RGB-only MAE (He et al., 2021) exhibit similar specialization trends.

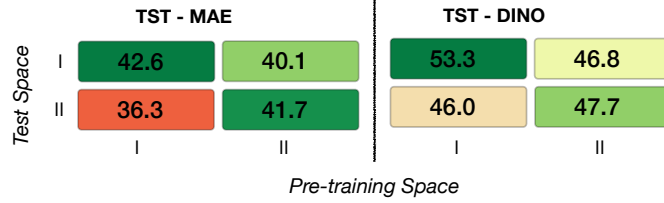


Figure 19: **Specialization using other objectives.** We further demonstrate the specialization using other pre-training objectives including MAE and DINOv2. The results shows similar specialization trend considering the other pre-training objectives.

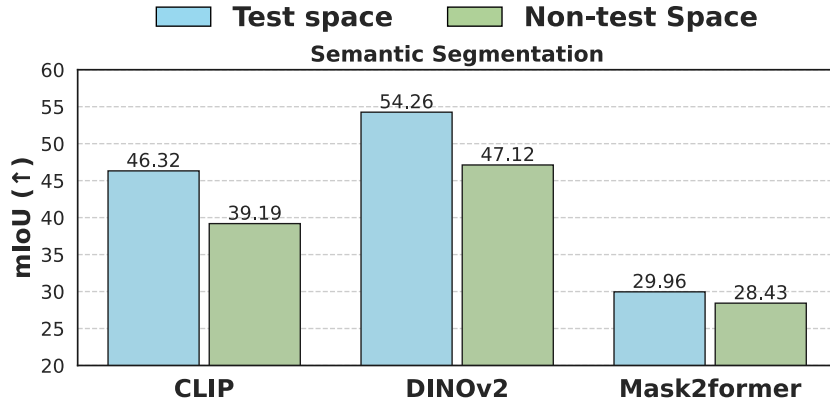


Figure 20: **Distillation in test space.** We find distilling over data from the test space, from various off-the-shelf models, results in more performant models in the test space. All results here are with the ViT-B backbone, on ProcTHOR (Deitke et al., 2022).

## R IS DISTILLING IN THE TEST SPACE BENEFICIAL?

As discussed in Sec. 4.5, we scale modalities by pseudo-labelling RGB data with various Internet-based models (Oquab et al., 2023; Radford et al., 2021; Cheng et al.). This process of creating additional modalities, and pre-training on them has enabled powerful multimodal foundation models (Mizrahi et al., 2023; Bachmann et al., 2024; 2022). This form of pre-training can also be seen as distilling the knowledge from these powerful off-the-shelf networks into a single unified model. With TST-MM, we also distill from various off-the-shelf networks like CLIP (Radford et al., 2021), DINOv2 (Oquab et al., 2023) with masked modelling (He et al., 2021; Mizrahi et al., 2023). Results in Tab. 2 suggest that distilling with multiple modalities on the test space, results in performant specialist models. However, to disentangle the effect of multimodality and distillation, we take it one step further to probe whether just distilling in the test space, provides some additional benefit, over non-test spaces? Therefore, we distill, CLIP (Radford et al., 2021), DINOv2 (Oquab et al., 2023) and Mask2former (Cheng et al.) in the test space, and compare it with distilling in an IID, but non test space, and report the results in Fig. 20 on semantic segmentation in ProcTHOR (Deitke et al., 2022). We find that distilling over data from the test space is more performant than the data from non-test spaces, underscoring the importance of access to the test space for specialization.

## S APPLICATION FOR HARDWARE DATA COLLECTION

As discussed in Sec. 3.2, TST can be extended to leverage more hardware-based modalities, such as IMU, GPS, Audio, which can be found on most common user devices, such as iPhone. To facilitate

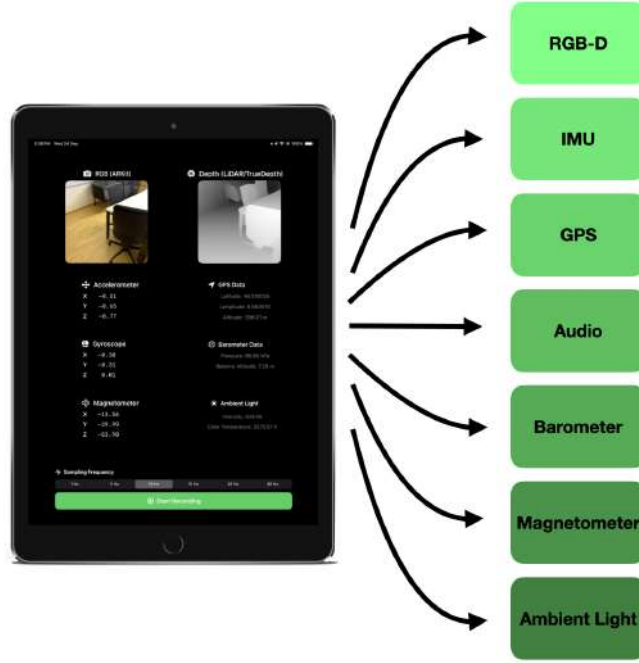


Figure 21: **iOS application for custom data collection.** The interface of the iOS application that allows collecting sensor data from any apple device with a camera.

future research in this area, we release an iOS application that enables anyone to collect aligned multimodal data from RGB-D and additional hardware sensors present on an iPhone. Fig. 21 shows an overview of our application.

## T TEST TIME TRAINING WITH TST

As noted in Sec. 2, we share a similar goal with Test-time Training (TTT) (Sun et al., 2020) in bridging the train-test divide. TTT does it from the lens of inference time optimization to specialize to a particular test instance, whereas TST attempts to specialize to a given test space by pre-training in it.

However, in practice, these strategies can be orthogonal and complement each other. We can potentially apply TTT to a model pre-trained with TST, to improve its performance. To benchmark how this combination works, we conduct an analysis where we apply Test-time training with masked autoencoders (TTT-MAE) (Gandelsman et al., 2022a), with two pre-trained methods, MAE (He et al., 2021) pre-trained on Internet data and TST-MAE pre-trained on the test space.

In TTT-MAE, we first start with a pre-trained MAE ViT-B encoder as the backbone, and train a task-specific head on the transfer set. During the test phase, the backbone is further tuned using the masked modeling objective for each test sample individually. This adaptive tuning enhances the model’s performance on the downstream task for the given test samples.

As presented in Tab. 8, TTT-MAE improves the mIoU results for both the Internet pre-trained backbone and TST-MAE. However, we find that TST-MAE gets significantly more improvement than Internet-based MAE (He et al., 2021). Both models use a ViT-B backbone and are tested on the semantic segmentation on the ProcTHOR (Deitke et al., 2022) dataset.

## U CONTINUAL LEARNING WITH TST.

As discussed before, when pre-training is performed on the exact same test space we deploy on, TST results in the most performant models. However, TST specializes in the test space, and all its

MAE (He et al., 2021) pre-training	Before TTT (mIoU $\uparrow$ )	After TTT (mIoU $\uparrow$ )
Internet	34.54	39.28
TST	<b>35.48</b>	<b>42.41</b>

Table 8: **TST with Test-Time Training.** Before TTT corresponds to the performance of the models directly after the transfer training without any test-time training, whereas after TTT shows the results when test-time training on the test samples is performed. Both models use a ViT-B backbone and are evaluated on the semantic segmentation on ProcTHOR (Deitke et al., 2022).

characteristics, at the state when the pre-training data was collected. Therefore, a natural question to ask is, what happens if the test space undergoes some changes after data collection? This could include changes in the lighting of the space or minor object placements. We begin by investigating if these changes lead to a drop in performance for the TST model trained on the original test space. Thereafter, we leverage the ability of ProcTHOR to randomize object placements and lighting to create a perturbed version of the test space. Note that the overall layout and assets remain exactly the same, only the lighting and placement of small objects are varied.

We first evaluate the performance of TST-MM pre-trained on the unperturbed test space, on the perturbed test space (Fig. 22, right), and we find that it experiences a drop as compared to its performance in the original test space, (Fig. 22, left). However, as we continually pre-train the model by collecting data in the updated test space (TST-MM (CL)), it quickly recovers the loss in performance, and is still highly performant as compared to Internet-based generalists (Bachmann et al., 2024). This suggests that even under the condition that the test space undergoes changes, by simply continuing data collection in the test space, TST can continually improve its performance, without any access to external data.

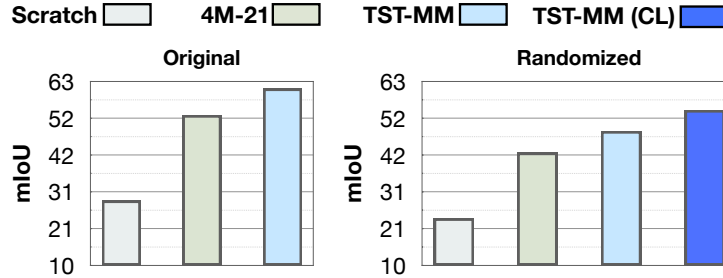


Figure 22: **Continual Learning with TST.** We study the performance of TST-MM, as the test space, undergoes lighting and minor object placement changes. The plot on the left, shows the result of the baselines on the original test space, without any changes. On the right, we present results after the test space has undergone lighting and object displacements. As expected, the TST-MM trained in the original test space, loses some performance, however as we continually train by collecting pre-training in the perturbed test space, we find that TST-MM (CL) quickly recovers performance.

## V SAMPLING RATIO BETWEEN TEST SPACE AND TRANSFER DATA DURING TST PRE-TRAINING

As mentioned in Section 4.1, we found mixing RGB images from the transfer set to our pre-training data beneficial for performance. To study the interplay of this dataset mix further, we analyze the effect of the sampling frequency of the samples from the transfer set and the test space data during pre-training. A ratio of 1/1 implies that half the samples in pre-training come from the test space data and the other half from the transfer dataset. We pre-train the TST-MM model using both small and base sizes on the same test space as in Tab. 2, in the ProcTHOR (Deitke et al., 2022) dataset under different ratios. The models are then transferred and evaluated on the semantic segmentation task. Tab. 9 demonstrates the results for various ratio configurations and their effect on different model sizes. First, we find that in all cases, TST-MM consistently outperforms Internet-based 4M-21 (Bachmann

et al., 2024) models of the same size. Secondly, we note that the performance of the bigger ViT-B based models is not sensitive to the ratio of sampling transfer and test space data, whereas for smaller ViT-S based models, a ratio of 1/1 seems to be a reasonable default choice.

Model Size	Transfer set / Test space set sampling ratio				4M-21
	1/1	1/4	1/8	1/16	
Small	<b>61.01</b>	59.03	56.96	57.01	46.12
Base	60.36	60.65	<b>60.85</b>	60.36	53.24

Table 9: **The effect of the sampling ratio between the test space and transfer data during pre-training.** We report transfer performance on semantic segmentation as we vary the sampling ratio between transfer and test space data during pre-training. There is no significant difference in results across different ratios for the base model, and for the small model, the best result is obtained with a one-to-one sampling ratio between the transfer set and the test space set. Irrespective of the sampling ratio observe TST-MM models always outperform Internet-based 4M-21 pre-training (Bachmann et al., 2024)

## W EXPERIMENTAL SETUP DETAILS

### W.1 PRE-TRAINING DETAILS

**Initialization.** For TST-MM, we use two initializations for pre-training. Unless stated otherwise, we pre-train our model from scratch, following the hyperparameters in Tab. 10. Additionally, for adaptation results in Tab. 2, we start from a pre-trained 4M-21 (Bachmann et al., 2024) model and finetune it with the hyperparameters in Tab. 11.

**DINO Pre-training.** For the DINO TST pre-training in Sec. Q, we use the implementation from the original DINOv2 repository<sup>2</sup>. We use the default provided training configuration files and train a model with the ViT-B/14 backbone for 300,000 steps with a batch size of 1024.

Configuration	Small	Base
Training length ( $n$ tokens)	100B	500B
Warmup length ( $n$ tokens)	10B	
Optimizer	AdamW (Loshchilov & Hutter, 2019)	
Opt. momentum	$\beta_1, \beta_2 = 0.9, 0.95$	
Base learning rate (Goyal et al., 2017)	1e-4	
Batch size	4096	
Weight decay	0.05	
Learning rate schedule	Cosine decay	
Feedforward activation	SwiGLU (Shazeer, 2020)	
Input token budget	128	256
Target token budget	128	256
Input and target $\alpha$	Mixture (Bachmann et al., 2024)	
Masking strategy	Mixture (Bachmann et al., 2024)	
Image resolution	224 <sup>2</sup>	
Augmentation	Random Crop	
Repeated sampling (Feichtenhofer et al., 2022)	4	
Data type	bfloat16 (Burgess et al., 2019)	

Table 10: **Pre-training settings for scratch initialization.** Training configuration for TST-MM initialized from scratch.

<sup>2</sup><https://github.com/facebookresearch/dinov2>



Configuration	Small	Base
Training length ( $n$ tokens)	100B	
Warmup length ( $n$ tokens)	10B	
Optimizer	AdamW (Loshchilov & Hutter, 2019)	
Opt. momentum	$\beta_1, \beta_2 = 0.9, 0.95$	
Base learning rate (Goyal et al., 2017)	5e-5	
Batch size	4096	
Weight decay	0.05	
Learning rate schedule	Cosine decay	
Feedforward activation	SwiGLU (Shazeer, 2020)	
Input token budget	128	256
Target token budget	128	256
Input and target $\alpha$	Mixture (Bachmann et al., 2024)	
Masking strategy	Mixture (Bachmann et al., 2024)	
Image resolution	224 <sup>2</sup>	
Augmentation	Random Crop	
Repeated sampling (Feichtenhofer et al., 2022)	4	
Data type	bfloat16 (Burgess et al., 2019)	

Table 11: **Pre-training settings for Internet initialization.** Pre-training configuration for TST starting from the the pre-trained 4M (Bachmann et al., 2024) model weights.

## W.2 TRANSFER DETAILS

**Semantic segmentation:** For semantic segmentation on ProcTHOR (Deitke et al., 2022), Replica (Straub et al., 2019) and Scannet++ (Yeshwanth et al., 2023) datasets, we use the ViT encoder from the pre-trained models with a decoder head, based on the ConvNext (Liu et al., 2022) network with a depth of 4. This decoder head is initialized from scratch. Training details are provided in Tab. I2. On Replica (Straub et al., 2019), and ProcTHOR (Deitke et al., 2022), pre-trained models are transferred and evaluated using a transfer training dataset of 20,000 images and evaluated on 5000 images sampled from the test space. On Scannet++ (Yeshwanth et al., 2023), we use a transfer dataset of 40,000 images and evaluated on 3000 images from the test space.

Configuration	Small	Base
Fine-tuning epochs	64	
Warmup epochs	1	
Optimizer	AdamW (Loshchilov & Hutter, 2019)	
Opt. momentum	$\beta_1, \beta_2 = 0.9, 0.999$	
Learning rate	1e-4	2e-4
Batch size	32 (16 for Scannet++)	
Weight decay	0.05	
Learning rate schedule	Cosine decay	
Layer-wise lr decay (Clark et al., 2020)	0.75	
Drop path (Huang et al., 2016)	0.1	
Input resolution	224 <sup>2</sup>	
Augmentation	RandomFlip + RandomCrop	

Table 12: **Semantic segmentation settings.** Configuration used for fine-tuning the pre-trained models on the semantic segmentation task.

**Object detection.** For object detection, we evaluate pre-trained models by using the ViT-based pre-trained encoder as the feature extractor in the detection framework. We use Cascade Mask-RCNN (He et al., 2017; Cai & Vasconcelos, 2017) as our primary object detection model. Besides the feature extractor, the other learnable components including the detector’s neck and head are

initialized from scratch. All training and evaluations are performed using the Detectron2 (Wu et al., 2019) framework. Exact training settings are provided in Tab. 13. We evaluate object detection in the test spaces from the ProcTHOR (Deitke et al., 2022) dataset as described in Section 4.1. For transfer, we use a dataset of 20,000 images from an external space, that is different from the test space. We evaluate the transferred model on 5000 images from the test space.

Configuration	Small	Base
Fine-tuning epochs	150	
Optimizer	AdamW (Loshchilov & Hutter, 2019)	
Opt. momentum	$\beta_1, \beta_2 = 0.9, 0.999$	
Weight decay	0.1	
Learning rate	0.0001	
Learning rate schedule	Multi-step decay	
Lr schedule milestones	[Epoch 133, Epoch 144]	
Lr schedule decay values	[1.0, 0.1, 0.01]	
Warmup epochs	0.01	
Batch size	128	
Layer-wise lr decay (Clark et al., 2020)	0.7	
Drop path (Huang et al., 2016)	0.1	
Input resolution	224 <sup>2</sup>	
Augmentation	RandomFlip + RandomCrop	

Table 13: **Object detection settings.** Configuration used for fine-tuning the pre-trained models on the object detection task.

**Image Captioning.** For image captioning, we evaluate the pre-trained models obtained from various methods including TST, 4M-21 (Bachmann et al., 2024) (Internet), and also include randomly-initialized baselines (training from scratch). We adopt a standard transformer-based encoder-decoder architecture for image captioning and employ cross-entropy loss for next-token prediction during training. Images are input to the encoder which serves as the context for the decoder network. The encoder network is initialized from the respective method’s encoder while the decoder is initialized randomly. Training and hyperparameter details are listed in Tab. 14. We also train a LLaVA style (Liu et al., 2023) model that serves as a Large-language-model-based baseline. We first train the connector module (MLP layer) using LLaVA’s first-stage pretraining data consisting of 558K image-text pairs subset of the LAION-CC-SBU dataset (et al., 2021). For second-stage, we re-format our ProcTHOR captioning dataset into instruction-tuning format and jointly finetune both the connector and LLM.

**Captioning data generation.** To train models on the captioning task, we create a transfer dataset on a set of external spaces by generating captions using GPT-4o (OpenAI, 2023) for the transfer dataset. We follow a similar procedure for the evaluation set from the test space. We ensure the quality of generated captions by providing GPT-4o with multi-modal inputs that include i) original RGB image ii) RGB image with instance-wise detection boxes and class names overlaid iii) Class names and bounding box coordinates in text format. We design an input prompt that instructs GPT-4o to leverage the multi-modal inputs and generate COCO-style (Lin et al., 2014) 5 concise captions with global context per image. For a sanity check, we randomly sampled 500 generated samples from the transfer set and found all captions to be consistent with the visual contents present in their respective images. The prompt message used for generating captions from GPT-4o is shown in Fig. 23.

## X COMPUTATIONAL RESOURCES.

All model pre-training and adaptations were done on 64 H100 GPUs, with the base and small models taking approximately 12 hours and 7 hours to train, respectively. For the semantic segmentation transfer runs, we fine-tuned the models on 4 H100 GPUs, resulting in approximately 3 hours of training for the base model and 1.5 hours for the small model. For the detection task, we only fine-tuned the base model on 8 A100 GPUs, training for approximately 6 hours. Similar to detection, for captioning we only fine-tuned the base model training on 8 H100 GPUs for approximately 6 hours.

I have a dataset of images captured in indoor settings showcasing different common household objects. I want to create COCO-style concise and global captions for these images. Please generate a single caption for each image, adhering to the following guidelines:

- \*\*Global Context but Concise\*\*:**  
The caption should be objective, describing the prominent objects and their spatial relationships within the scene. Each caption must cover the global scene context and prominent objects.
- \*\*Use of Ground-Truth Classes\*\*:**  
Along with each image, ground-truth classes and bounding box information are provided. Bounding box information is in the format `(upper left x coordinate, upper left y coordinate, width, height)`. Use bounding box information for correct spatial relationships (such as left side, right side, top, below, etc.) between objects.
- \*\*Bounding Boxes and Class Labels Visualized in Image\*\*:**  
The bounding boxes and class names are overlaid on the image, showing each detected class for better localization.
- \*\*Spatial Positioning\*\*:**  
Describe all objects' positions and spatial relationships as visible in the image and ground-truth information to help locate them accurately. If multiple objects are present in the image (as indicated in ground-truth information), explicitly mention their count and explain their positional relationships with other objects in the image.
- \*\*No Hallucinations!\*\***  
Each generated concise caption must agree with the actual contents shown in the provided image. Strictly avoid adding information about objects unless you are certain. Only utilize the information visible in the image and the provided ground-truth class information.

I will provide both the original image and the image with overlaid boxes and labels. Use both images to provide a grounded global and COCO-style concise caption.

**\*\*Format your response\*\*:**  
Return a Python list containing concise global captions. Do not output any other text.

Ground Truth information: `GT_class_and_bbox_information`  
Image with Annotations: `Image_Annotated`  
Original Image: `Image_Original`

Figure 23: **LLM Prompt instruction for ProcTHOR caption generation transfer task.** We generate ground-truth captions by providing multi-modal information to GPT-4o (OpenAI, 2023) including annotated image, class and instance-wise bounding-box information. For each image, we generate 5 COCO-style captions.



Figure 24: **Additional qualitative results.** As demonstrated here TST performs better compared to the other models for all tasks.

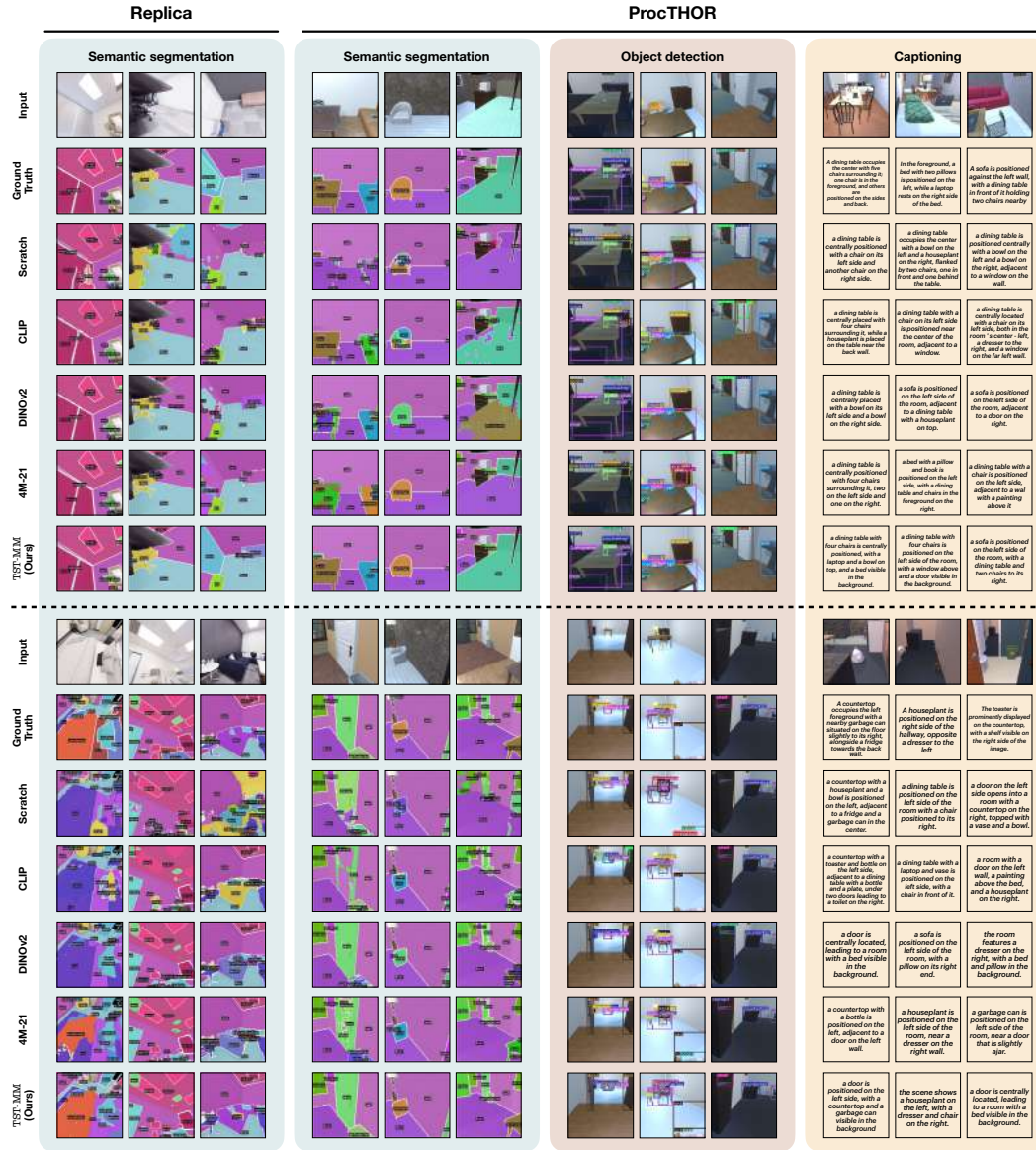


Figure 25: **Additional qualitative results.** As demonstrated here TST performs better compared to the other models for all tasks.

Configuration	ProcTHOR Captioning
Fine-tuning epochs	1400
Warmup epochs	600
Optimizer	AdamW (Loshchilov & Hutter, 2019)
Opt. momentum	$\beta_1, \beta_2 = 0.9, 0.95$
Base learning rate (Goyal et al., 2017)	1e-5
Batch size	2048
Weight decay	0.05
Learning rate schedule	Cosine decay
EMA decay	SwiGLU (Shazeer, 2020)
Eval. freq (epochs)	50
Input resolution	224

Table 14: **Training details: Image Captioning.** Configuration used for transfer training for image captioning.