SSL Enables Learning from Sparse Rewards in Image-Goal Navigation

Arjun Majumdar [1 2] **Gunnar A. Sigurdsson** [1] **Robinson Piramuthu** [1] **Jesse Thomason** [1 3] **Dhruv Batra** [2]
Gaurav S. Sukhatme [1 3]

Abstract

Visual navigation through novel environments is an essential skill for intelligent household robots. Reinforcement learning (RL) with dense reward shaping is the standard approach for such visual navigation problems. Dense rewards are used despite the fact that they penalize exploration—a key skill for navigating in new environments—because of the commonly held belief that sparse rewards do not work. In this paper, we present a surprising finding. We demonstrate that combining sparse rewards with self-supervised learning (SSL) not only makes them work, but also *outperforms dense rewards*, which is the first result of this kind. We apply our approach alongside data augmentation to train a general-purpose agent for image-goal navigation (ImageNav), in which the goal is presented as an image captured from the target location. These techniques improve navigation success from 64% to 72% (+8), path efficiency—measured by success weighted by path length (SPL)—from 0.50 to 0.62 (+0.12), and also lead to a +0.06 absolute improvement in SPL over the state-of-the-art.

1. Introduction

Learning to navigate in new indoor environments using only visual observations and without a map is a key challenge for general-purpose household robots. In image-goal navigation (ImageNav) (Zhu et al., 2017), agents perform such visual navigation to find the location where a target image was captured (Figure 1). For agents to *efficiently* perform image-goal navigation, rather than fall back on exhaustive search, they need to learn reliable ways to exploit the structural priors of indoor spaces, such as that stoves are in kitchens and that sofas are often found in living rooms but not in kitchens. However, learning to correlate these visual cues in a robust and generalizable manner remains an open problem.

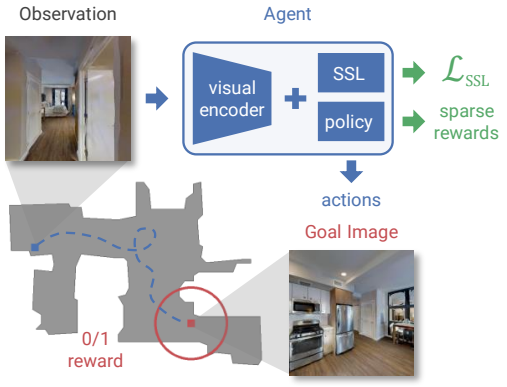

Figure 1. In ImageNav, agents must find where a goal image was captured using only visual observations and without a map. We discover that combining self-supervised representation learning and sparse rewards can work surprisingly well—outperforming *dense reward* alternatives and setting a new state-of-the-art.

Reinforcement learning (RL) is the standard tool used for visual navigation problems. Current methods (e.g., Mezghani et al. 2021; Kwon et al. 2021) use dense rewards because the community believes (understandably so) that learning with sparse rewards does not work. Indeed, considerable effort in RL focuses on designing complex reward functions, often with 10 or more terms (e.g., Kumar et al. 2021). However, reward engineering is hard and sometimes unsuccessful despite significant effort and training at scale, such as after 100M frames of experience as shown in Szot et al. (2021).

In the context of visual navigation, the most commonly used dense rewards encourage following shortest-paths. A shortest-path does not contain any exploration, but image-goal navigation requires exploration because the environment is new and the agent does not know the goal location, only an image. Unfortunately, exploration increases distance travelled; thus, this reward penalizes exploration.[1] In this paper, we present a surprising finding. Rather than solve this problem through reward engineering, as done in Maksymets et al. 2021, we show that combining sparse rewards with self-supervised learning (SSL) allows them to

[1]Amazon Alexa AI [2]Georgia Institute of Technology [3]University of Southern California. Correspondence to: Arjun Majumdar <arjun.majumdar@gatech.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

[1]A similar argument is made in Maksymets et al. (2021).

not only work, but also outperform *dense rewards*, which is the first result of this kind.

We identify two ingredients that make learning image-goal navigation with sparse rewards work: long training schedules and self-supervised learning. Intuitively, using sparse rewards increases the difficulty of the learning problem. Indeed, we find significantly lower performance—as measured by success weighted by path length (SPL)—early in training ($<$50M frames). Interestingly, we discover the emergence of a curriculum, with better sparse reward performance first on short goals less than 3m away after $\sim$60M frames and medium 3-5m goals after $\sim$100M frames. However, we also find that simply training longer does not lead to better SPL or navigation success rates on goals $>$5m away.

We hypothesize that a key difficulty is learning strong visual representations from scratch using only sparse rewards because there is no semantic supervision (e.g., image labels or bounding box annotations). Thus, it is difficult to learn generalized representations of what, for example, stoves or kitchens look like. A straightforward approach for improving the visual representations is to train with data augmentation techniques from 2D computer vision. However, some augmentations increase the difficulty of the RL training task, as we find in our experiments.

We address this issue with an auxiliary loss adapted from 2D self-supervised visual representation learning (He et al., 2020; Chen et al., 2020b;a). Our loss is designed to improve the visual representations without increasing task difficulty. We use strong augmentations for representation learning and weak augmentations for reinforcement learning (Figure 2). We discover that decoupling these two learning problems is particularly useful for learning from sparse rewards, leading to better navigation performance than dense reward alternatives. We also find that this framework outperforms simply using strong augmentations directly in the RL pipeline.

We evaluate our ideas on an ImageNav task in which agents only have access to RGB sensor inputs. This vision-only setting allows studying how ideas from 2D computer vision translate without introducing confounding factors, such as language processing for tasks like vision-and-language navigation (VLN) (Anderson et al., 2018b).

Contributions. A summary of our contributions are:

- We demonstrate that a general-purpose agent trained with SSL and sparse rewards can outperform agents trained with dense reward shaping. This surprising finding runs counter to intuition built from years of experimental evidence.
- We improve the navigation performance of a general-purpose agent in terms of success rate (SR) from 64% to 72% (+8%) and success weighted by path length (SPL) from 0.50 to 0.62 (+0.12).
- We demonstrate that a general-purpose agent can outperform the previous state-of-the-art—achieved by a model with a navigation-specific architecture (Mezghani et al., 2021)—by +0.06 in SPL.
- We demonstrate that training with SSL plus sparse rewards results in richer visual representations as measured by a proxy indoor scene classification task, and see a +6.4% absolute improvement in top-1 classification accuracy over our baseline agent.

Source code and pretrained models will be publicly released.

2. Related Work

Visual Navigation. A number of visual navigation tasks require exploring new environments to reach goals that can be specified as objects (Batra et al., 2020), rooms (Anderson et al., 2018a), images (Zhu et al., 2017), or natural language instructions (Anderson et al., 2018b). This work focuses on image-goal navigation using only RGB sensors, which allows us to study transferring techniques from 2D computer vision in isolation. Furthermore, we use a general-purpose agent with the expectation that the findings will transfer.

Image-Goal Navigation. Significant progress has been made on the image-goal navigation task (Zhu et al., 2017; Chaplot et al., 2020b;c; Mezghani et al., 2021; Kwon et al., 2021). Modular architectures construct metric or topological representations of the environment (Chaplot et al., 2020b;c). Memory mechanisms in agent architectures can maintain a sparse representation of an agent's observation history (Mezghani et al., 2021; Kwon et al., 2021). These tactics incorporate navigation-specific inductive bias into model architectures. By contrast, we use a general-purpose agent architecture, and focus on training without dense reward shaping, which penalizes exploration at test-time.

Reward Shaping in Visual Navigation. Dense reward shaping is a standard technique for visual navigation tasks. In ImageNav, prior work (Mezghani et al., 2021; Kwon et al., 2021) uses the reward from Savva et al. (2019), which includes dense reward shaping based on the per timestep change in the distance to the goal. A similar reward structure has been used for object-goal navigation (Chaplot et al., 2020a) and instruction-guided navigation (Tan et al., 2019). To improve exploration and navigation performance, alternative reward shaping designs have been explored for object-goal navigation (Ye et al., 2021; Maksymets et al., 2021) and instruction-guided navigation (Hong et al., 2021). Reward shaping is thought to give higher accuracy at the cost of reward engineering and diminished exploration.We show that this trade-off is unnecessary: generalizable exploration behaviors can be learned by combining sparse rewards with self-supervised representation learning.

Self-Supervised Learning in RL. Motivated by the success

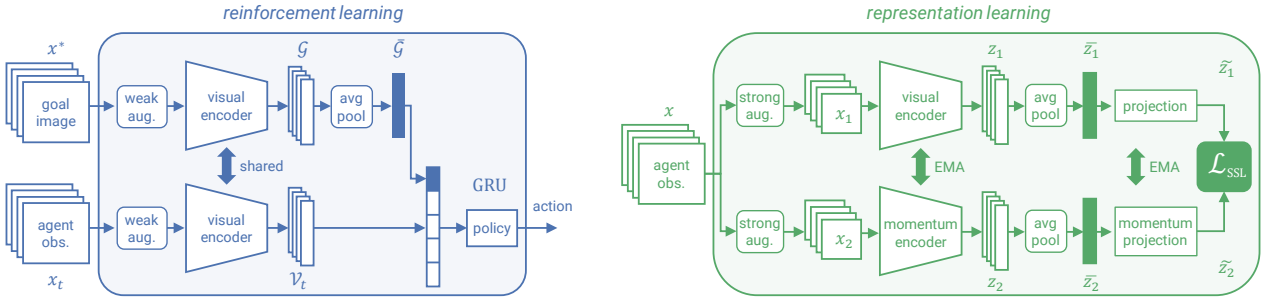

Figure 2. **Overview of our approach.** Our framework combines RL using weak data augmentation (left) with SSL using strong augmentation (right). Our ImageNav agent (left) processes panoramic observations x_t and goal images x^* with a shared visual encoder, and then processes the concatenated outputs $[\mathcal{V}_t, \bar{\mathcal{G}}]$ with a policy network that predicts actions. For representation learning (right), we transform a panoramic observation x with two strong augmentations. One view is processed by the visual encoder used for RL and then average pooled and projected to produce $\tilde{z}_1$. The other view is similarly processed with networks updated as an exponential moving average (EMA) of their top-branch counterparts. We jointly train with a contrastive loss $\mathcal{L}_{\text{SSL}}$ and RL training objectives.

of self-supervised visual representation learning in 2D vision (Misra & van der Maaten, 2020; He et al., 2020; Chen et al., 2020a), recent methods have used self-supervised learning within RL frameworks. Laskin et al. (2020b) introduce a simple framework that combines contrastive learning and RL training objectives, Hansen et al. (2021) use self-supervised learning to adapt policies at test-time, and Stooke et al. (2021) decouple representation learning from RL training. Hansen & Wang (2021) pull the representations of augmented data closer to representations of non-augmented data, while jointly training an RL policy without any data augmentation. By contrast, our framework pulls representations from two strong augmentations closer, and uses weak augmentations within the RL pipeline. This asymmetric design fully leverages self-supervised learning, and uses weak augmentations to improve navigation performance. We study self-supervised learning in the context of learning from sparse vs. dense rewards, and discover that representation learning is more useful without dense reward shaping.

3. Task Description

In image-goal navigation (ImageNav) an agent must move (without a map) from a starting location to a goal communicated as an image (Figure 1). For fair comparisons with prior work (Chaplot et al., 2020c; Mezghani et al., 2021), we consider a setting in which the goal image and agent's observations come from a panoramic RGB sensor, and no other sensor data is available (e.g., no depth camera).

Observation space. At each timestep t the agent receives a panoramic image x_t and has access to a panoramic goal image x^*. Each panorama consists of four 128×128 RGB frames collected at regular angles (0°, 90°, 180°, and 270°).

Action space. The agent can select one of four actions: `MOVE_FORWARD` by 0.25m, `TURN_LEFT` by 10°, `TURN_RIGHT` by 10°, and `STOP`.

Success Criteria. Navigation is considered successful if the `STOP` action is selected when the agent is within 1m of the goal position in terms of geodesic distance.

4. Method

This section describes the general-purpose agent (Section 4.1) that we train using model-free RL with sparse rewards (Section 4.2). We also describe the weak data augmentations used in the RL training pipeline (Section 4.3) and the strong augmentations used for SSL (Section 4.4).

4.1. General-Purpose Agent

We use an agent architecture consisting of a visual encoder and a policy network, illustrated in Figure 2 (left). The goal image and agent observations are processed by the visual encoder, and the encoded outputs are processed by the recurrent policy network that maintains the agent's state and predicts actions. We use a CNN-based visual encoder and a single-layer GRU as our policy network.

At each timestep t, the agent encodes the panoramic *observation* x_t with a CNN to produce average-pooled visual features $\mathcal{V}_t \in \mathbb{R}^{4\times d}$ corresponding to the 4 RGB frames in each panorama and a CNN with d output feature channels. Similarly, the panoramic *goal image* x^* is encoded with the same CNN producing features $\mathcal{G} \in \mathbb{R}^{4\times d}$. We average-pool the goal image features from the 4 RGB frames to produce a d dimensional goal representation $\bar{\mathcal{G}}$, which follows the approach of Mezghani et al. 2021. This average-pooling operation makes the goal representation invariant to the orientation of the goal viewpoint, which matches the task specification that requires stopping near the goal at any orientation. The visual features $\mathcal{V}_t$ are concatenated with the goal representation $\bar{\mathcal{G}}$ to produce the input to the recurrent

policy network, in our case a GRU:

$$h_t = \text{GRU}\left(\left[\mathcal{V}_t, \bar{\mathcal{G}}\right], h_{t-1}\right). \quad (1)$$

The GRU output h_t is passed to a linear layer that generates a softmax distribution over the action space.

This agent is structurally similar to the agent from Ramakrishnan et al. 2021; Wijmans et al. 2019, which is used for the related point-goal navigation task, and achieves 100% point-goal success on the Gibson dataset (Xia et al., 2018). The primary difference is in the goal representation, which is derived from a `GPS+Compass` sensor in point-goal navigation. This architectural resemblance is intentional, and highlights the generality of this agent architecture.

Dense Reward Shaping. In visual navigation, it is common to use the reward function from Savva et al. 2019, which is composed of a sparse success reward r_{success}, a dense reward shaping term based on the change in the distance to the goal Δ_{dtg}, and a slack penalty r_{slack}. Specifically, the reward is:

$$r(s_t, a_t) = r_{\text{success}} - \Delta_{\text{dtg}} + r_{\text{slack}}\,, \quad (2)$$

where $r_{\text{success}} = 2.5$ if the selected action a_t is `STOP` and the agent's state s_t is within 1m of the goal and 0 otherwise, Δ_{dtg} is calculated as the per step change in the geodesic distance to the goal, and $r_{\text{slack}} = -0.01$ encourages path-efficiency. The Δ_{dtg} term provides a dense signal that rewards actions on the shortest path to the goal and penalizes actions taking the agent geodesically further from the goal.

4.2. Learning from Sparse Rewards

Our primary observation is that the dense reward shaping term Δ_{dtg} is problematic because it encourages navigating along the shortest path, which can lead to memorizing training environments to maximize the reward as demonstrated by (Maksymets et al., 2021). However, rather than add additional reward terms that explicitly encourage exploration, as done by Maksymets et al. 2021, we opt for a more elegant solution. We drop the reward shaping term Δ_{dtg} altogether, and train our agent with the sparse reward:

$$r(s_t, a_t) = r_{\text{success}} + r_{\text{slack}}\,. \quad (3)$$

Intuitively, sparse rewards make the learning problem more challenging to solve. Indeed, in our experiments (Section 5) we observe that in the early stages of training agents struggle to learn from this sparse signal. However, we identify two techniques that can be used to overcome this challenge: longer training schedules and self-supervised visual representation learning using strong data augmentations.

4.3. Weak Augmentations for RL

In this section, we describe the augmentations used for RL training. Specifically, we use two standard techniques from computer vision literature: color jittering and random cropping. These augmentations are identical to the ones used in Mezghani et al. (2021) allowing for direct comparisons of the two methods. Color jittering randomly adjusts the brightness, contrast, saturation, and hue of an image. Random cropping extracts a region from an image at a random scale and aspect ratio, and then resizes it to the fixed size expected by the visual encoder. Specifically, we use randomly sampled scaling factors between 0.8 to 1.0, which is smaller than the range typically used in 2D computer vision for supervised (0.08, 1.0) or self-supervised learning (0.2, 1.0). Thus, herein, we refer to this form of random cropping as spatial jittering. Collectively, we refer to the combination of spatial and color jittering as *weak augmentations*.

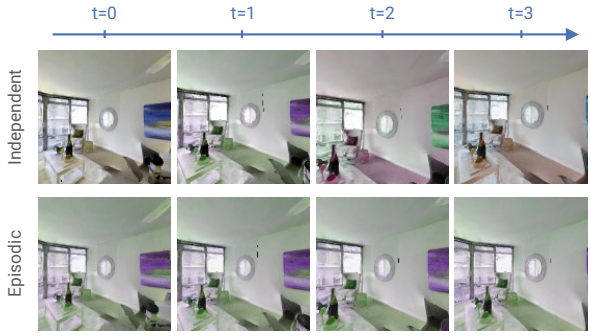

Figure 3. **Comparison of data augmentation strategies.** Independently applying augmentations over time (top) results in unrealistic variations such as the painting and flooring changing colors at each timestep. Uniformly applying the same, randomly sampled augmentation across time (bottom) fixes this issue.

Episodic vs. Temporally-Inconsistent Augmentations. In 2D computer vision and in Mezghani et al. 2021 data augmentations are independently applied to data to increase the variations seen during training. However, we observe that using different augmentations over time introduces unrealistic changes that do not reflect the testing conditions.[2] This challenge is shown in Figure 3 (top). In this example, the color of the painting on the wall unnaturally changes from blue to purple to green over time. We instead uniformly apply a single randomly sampled augmentation across time as shown in Figure 3 (bottom). In this case, the painting remains purple over the course of navigation.

While conceptually simple, in experiments in Section 5 we find that this technique substantially improves image-goal navigation performance. Intuitively, this approach retains the benefits of data augmentation without introducing additional, unrealistic challenges to the reinforcement learning problem. Furthermore, uniformly applying augmentations over time can be efficiently implemented in PyTorch (Paszke et al., 2019), which leads to an $\sim$1.9x improvement in wall-clock training time in our experiments (Section 5).

[2] A similar argument is presented in Laskin et al. (2020a).

Table 1. **Image-goal navigation performance in Gibson environments**. Overall gains over our baseline agent (Row 4) are indicated for SR and SPL in parentheses. Rows 4-6 use dense rewards Δ_{dtg}, while Rows 7-8 use sparse rewards. Rows 3-4 use temporally-inconsistent data augmentation (INC.) and Rows 5-8 use episodic augmentation (EPI.). Rows 6 and 8 use SSL. Combining SSL + sparse rewards (Row 8) outperforms dense reward alternatives. Methods: 1 (Chaplot et al., 2020b), 2 (Chaplot et al., 2020c), and 3 (Mezghani et al., 2021).

	METHOD	DATA AUG.	AUX. LOSS	REWARD SHAPING	EASY SR ↑	EASY SPL ↑	MEDIUM SR ↑	MEDIUM SPL ↑	HARD SR ↑	HARD SPL ↑	OVERALL SR ↑	OVERALL SPL ↑
1	ACTIVE NEURAL SLAM[1]				0.63	0.45	0.31	0.18	0.12	0.07	0.35	0.23
2	NEURAL TOPOLOGICAL SLAM[2]				0.80	0.60	0.47	0.31	0.37	0.22	0.55	0.38
3	MEMORY-AUGMENTED RL[3]	INC.		Δ_{DTG}	0.78	0.63	0.70	0.57	0.60	0.48	0.69	0.56
4	GENERAL-PURPOSE AGENT BASELINE	INC.		Δ_{DTG}	0.73	0.57	0.65	0.51	0.55	0.42	0.64	0.50
5	EPISODIC AUG.	EPI.		Δ_{DTG}	0.76	0.61	0.68	0.55	**0.62**	0.49	0.69 (+0.05)	0.55 (+0.05)
6	EPISODIC AUG. + SSL	EPI.	SSL	Δ_{DTG}	0.79	0.67	0.70	0.58	0.59	0.47	0.69 (+0.05)	0.57 (+0.07)
7	EPISODIC AUG. − SHAPING	EPI.			0.80	**0.72**	0.66	0.58	0.57	0.49	0.68 (+0.04)	0.60 (+0.10)
8	EPISODIC AUG. + SSL − SHAPING	EPI.	SSL		**0.81**	0.71	**0.73**	**0.62**	**0.62**	**0.51**	**0.72** (+0.08)	**0.62** (+0.12)

4.4. Strong Augmentations with SSL

This section describes a self-supervised auxiliary loss that improves the visual representations learned by our agent. This loss is designed to enable the use of stronger augmentations than described above, without increasing the difficulty of the RL training problem. Specifically, we use augmentations from 2D self-supervised visual representation learning methods from Chen et al. (2020b;a) that include strong random cropping (scaling between 0.2 to 1.0), horizontal flipping, grayscaling, color jittering, and Gaussian blurring. We refer to this set of augmentations as *strong augmentations*.

As illustrated in Figure 2, we used strong augmentations for self-supervised learning and applied weak augmentations for RL training. Intuitively, our approach separates visual representation learning from RL. However, both branches use a shared visual encoder f_θ that is updated under both training objectives with stochastic gradient descent (SGD). For representation learning, following Chen et al. (2020b), we also use a momentum encoder f_ω with parameters ω that are updated as an exponential moving average (EMA) of the parameters θ from the visual encoder f_θ.[3]

Specifically, given a panoramic image $x \in \mathbb{R}^{4\times W\times H\times 3}$, we apply independently sampled strong augmentations to each of the 4 frames to produces two views x_1 and x_2 of the image. The views are encoded to produce features $z_1 = f_\theta(x_1)$ and $z_2 = f_\omega(x_2)$ and then averaged-pooled across the 4 RGB frames, resulting in $\bar{z}_1 \in \mathbb{R}^d$ and $\bar{z}_2 \in \mathbb{R}^d$. The average-pooled features are processed with learned projections g_θ and g_ω to yield $\widetilde{z}_1 = g_\theta(\bar{z}_1)$ and $\widetilde{z}_2 = g_\omega(\bar{z}_2)$.[4] Finally, we use the InfoNCE (van den Oord et al., 2018) contrastive loss function as an auxiliary loss:

$$\mathcal{L}_{\text{SSL}} = -\log \frac{\exp(\widetilde{z}_1 \cdot \widetilde{z}_2/\tau)}{\exp(\widetilde{z}_1 \cdot \widetilde{z}_2/\tau) + \sum_{\widetilde{z}_n} \exp(\widetilde{z}_1 \cdot \widetilde{z}_n/\tau)}, \quad (4)$$

where $\widetilde{z}_n$ are sampled from other panoramas in the training batch and τ is a temperature parameter set to 0.07.

To summarize, in our approach, panoramic images are weakly augmented for RL training and strongly augmented for SSL. We jointly train with $\mathcal{L}_{\text{SSL}}$ and RL training objectives, specifically PPO (Schulman et al., 2015).

5. Experiments and Key Findings

Our experiments are focused on the following questions:

How do sparse rewards and SSL impact performance? We find that sparse rewards dramatically decrease performance early in training (<50M frames of experience), as might be expected (Figure 4). Longer training schedules lead to substantial improvements in path efficiency, but the success rate remains poor. Adding a self-supervised auxiliary loss fixes this issue, and on both metrics the SSL + sparse rewards solution outperforms dense reward variants. Specifically, we find that using SSL + dense rewards does not lead to similar gains (Table 1).

Do other ways of using strong augmentations work? We study using strong augmentations directly within RL training rather than through a self-supervised loss and find this alternative substantially decreases performance (Figure 5).

How does performance compare with state-of-the-art? Our general-purpose agent trained with SSL plus sparse rewards outperforms the state-of-the-art from Mezghani et al. 2021 in terms of success rate and path efficiency (Table 1).

Do the visual representations improve? We probe visual representations learned with different training objectives using an indoor scene classification proxy task, and find that

[3] Our approach is similar to Hansen et al. (2021), but importantly we use data augmentation in both the SSL and RL pipelines.

[4] The parameters of g_ω are also updated as an EMA of g_θ.

training with SSL + sparse rewards leads to the best representations, supporting our hypothesis that sparse rewards are better for learning visual representations (Table 2).

5.1. Implementation Details

Task Setup. We conduct experiments using the Habitat (Savva et al., 2019) simulator with 72 training and 14 test environments from the Gibson (Xia et al., 2018) dataset. In each episode the agent is allowed to take up to 500 actions. We report success rate (SR) and success weighted by inverse path length (SPL) (Anderson et al., 2018a), which is the product of binary success and the ratio between the length of the agent's path and the length of the shortest path to the goal. SPL provides a measure of path efficiency.

Image-Goal Navigation Dataset. We use training episodes generated by Mezghani et al. (2021) following the protocol from Chaplot et al. (2020c).[5] Specifically, for each training environment 3,000 episodes were randomly generated for three difficulty levels based on path length: EASY (1.5-3m), MEDIUM (3-5m), and HARD (5-10m), resulting in 9,000 episodes per environment. For evaluation, 100 episodes per difficulty level were generated for each test environment.

Training Setup. We train agents using PPO (Schulman et al., 2015) implemented in PyTorch (Paszke et al., 2019) and use DD-PPO (Wijmans et al., 2019) to efficiently distribute processing across 32 NVIDIA V100 GPUs.[6] We use a half-width ResNet18 (He et al., 2016) CNN as the visual encoder for processing agent observations and the goal image. RL methods are often trained only for 10-50M frames of experience (e.g., Du et al. 2021; Chaplot et al. 2020c), however we find that long-training schedules are particularly important. Thus, we train all models for 400M frames of experiences and use early stopping based on SPL. The experiments in Table 1 required over 18K GPU-hours for training. Additional details are included in the Appendix.

5.2. Impact on Navigation Performance

In Table 1, we study the effect of using sparse rewards and a self-supervised auxiliary loss with the general-purpose agent described in Section 4.1. Row 4 is the baseline performance of this agent. These baseline results are much stronger than the baseline from Chaplot et al. (2020c) who report 0.10 SPL overall as compared to our baseline of 0.50 SPL. This difference is in part because we train 16x longer (25M vs. 400M frames). Specifically, our baseline uses *weak augmentations* applied independently to each RGB frame, as done in Mezghani et al. (2021). This baseline is trained with dense reward shaping based on change in the distance-to-goal Δ_{dtg}, and does not use an auxiliary loss.

[5]Train and test data was received from Mezghani et al. (2021).

[6]We also use NVIDIA A40s for a subset of the experiments.

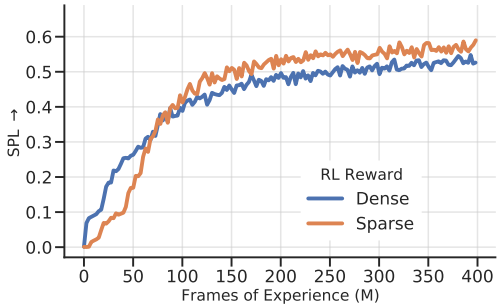

Figure 4. **Learning curves comparing dense vs. sparse rewards without SSL.** Dense rewards dominate before 50M, but after 100M frames using sparse rewards consistently results in higher SPL.

Episodic vs. Temporally-Inconsistent Augmentations. In Row 5 the data augmentations are applied uniformly over time. We compare this with the temporally-inconsistent augmentations used in our baseline and in the state-of-the-art model from Mezghani et al. (2021). We find two benefits of this simple technique. First, it improves SR from 64% to 69% (+5%) and SPL from 0.50 to 0.55 (+0.05) over the baseline (Rows 4 vs. 5). Second, uniform augmentations can be efficiently implemented in PyTorch (Paszke et al., 2019), which reduces the time required to train an agent for 400M frames of experience from ∼8.5 days to ∼4.5 days—a 1.9x improvement. Similar computational efficiencies were seen in Laskin et al. (2020a) with custom implementations. The remaining experiments uniformly apply augmentations over time (EPISODIC AUG.) due to the computational efficiency.

Learning with Sparse Rewards. In Row 7, we remove dense reward shaping (Δ_{dtg}) from the reward function. This removal improves SPL from 0.55 to 0.60 (+0.05) overall (Rows 5 vs. 7). In Figure 4, we plot learning curves in terms of SPL to compare the training behavior of dense vs. sparse rewards (Rows 5 and 7 in Table 1). We find early in training SPL scores are substantially lower with sparse rewards, and that over 100M frames of experience are required for SPL to improve. Thus, previous findings in RL reported for 10-50M frames may not fully represent learning behavior.

However, while SPL improves, particularly on EASY episodes from 0.61 to 0.72, we also see drops in SR on MEDIUM episodes from 68% to 66% and on HARD episodes from 62% to 57%, which leads to a slight drop in overall SR from 69% down to 68% (Rows 5 vs. 7). These performance drops reflect the difficulty in learning from sparse rewards alone, which we find are addressed by adding SSL.

SSL + Sparse Rewards. In Rows 6 and 8, we add the self-supervised auxiliary loss described in Section 4.4 to the RL training objective. We find SSL + sparse rewards (Row 8) outperforms training with dense rewards (Row 5) and SSL + dense rewards (Row 6). Specifically, compared with dense rewards, SR improves from 69% to 72% (+3%) and SPL from 0.55 to 0.62 (+0.07) overall (Rows 5 vs. 8). Com-

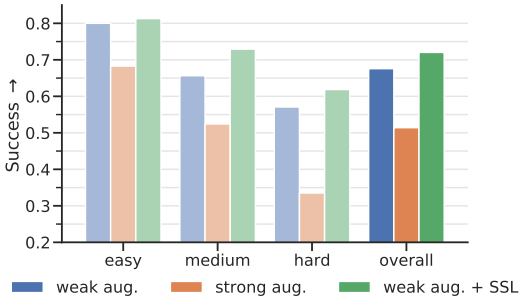

Figure 5. **Comparing methods for using strong augmentations.** Compared to using weak aug. for RL (blue), using strong aug. for RL (orange) substantially reduces SR, but our approach improves SR by using weak aug. for RL and strong aug. for SSL (green).

pared with SSL + dense rewards, SR improves from 69% vs. 72% (+3%) and SPL from 0.57 vs. 0.62 (+0.05) overall (Rows 6 vs. 8). Additionally, SSL + sparse rewards leads to improvements on all difficulty levels (EASY, MEDIUM, and HARD). In this setting, we find that representation learning is particularly effective with sparse rewards.

Cumulatively, applying augmentations uniformly over time and training with SSL + sparse rewards substantially improves SR from 64% to 72% (+8) and SPL from 0.50 to 0.62 (+0.12) over our baseline agent (Rows 4 vs. 8).

5.3. Using Strong Data Augmentation

We use strong augmentations within a self-supervised auxiliary loss. Alternatively, they could be directly used for RL training. In Figure 5, we study this alternative. Specifically, we use strong augmentation applied uniformly over time, and train with sparse rewards and without SSL (orange). We compare with sparse reward agents trained with weak augmentations (blue – from Table 1 Row 7) and weak augmentations + SSL (green – from Table 1 Row 8). We find that using stronger augmentations for RL dramatically reduces success rate (SR) from 68% to 51% (-17%) overall (blue vs. orange). By contrast, training with weak augmentations in RL and strong augmentation with SSL improves SR from 68% to 72% (+4%) overall (blue vs. green).

5.4. Comparison with State-of-the-Art

In this section, we compare with published results from prior work that also studies the RGB-only ImageNav setting. These results provide a comparison between models designed with navigation-specific inductive bias and our solution, which uses a general-purpose agent architecture.

- **Active Neural SLAM** Chaplot et al. (2020b) is a hierarchical approach for visual exploration combining a learning-based SLAM module with classical planning.
- **Neural Topological SLAM** Chaplot et al. (2020c) builds a topological graph-based representation of the environment to facilitate visual navigation.

Table 2. **Comparison of learned visual representations.** We evaluate visual representations using a linear probe on indoor scenes classes from Places365 (Zhou et al., 2017). Results averaged over 3 evaluation trials. Row numbers match Table 1. Improvements over our baseline (Row 4) are indicated for Top1 Acc in parentheses. We find SSL + sparse rewards (Row 8) is substantially better for representation learning than SSL + dense rewards (Rows 6).

	Method	Top5 Acc ↑	Top1 Acc ↑	
	Randomly Initialized	41.1	14.5	
4	General-Purpose Agent Baseline	55.0	24.2	
5	Episodic Aug.	56.0	24.7	(+0.5)
6	Episodic Aug. + SSL	60.7	28.7	(+4.5)
7	Episodic Aug. − Shaping	56.4	25.1	(+0.9)
8	Episodic Aug. + SSL − Shaping	**62.1**	**30.6**	(+6.4)
	ImageNet Pretraining	84.3	52.0	

- **Memory-Augmented RL** Mezghani et al. (2021) augments a general-purpose agent (similar to ours) with an external memory designed to retain a sparse, geographically separated set of previous state embeddings.

The results in Table 1 demonstrate that our full approach (Row 8) outperforms specialized models from prior work (Rows 1-3). Specifically, compared to the previous state-of-the-art (Row 3) we see a +0.06 improvement in SPL and a +3% improvement in success rate (SR). Furthermore, we see improvements in SR and SPL across all difficulty levels (EASY, MEDIUM, and HARD). We observe a large gain on EASY paths (+0.08 improvement in SPL), which suggests that our agent is able to learn behaviors that generalize better to new environments even in the simplest version of the task.

5.5. Impact on Visual Representations

Ideally, learning to locate image-goals in indoor environments should produce visual representations that can differentiate rooms in a house, such as a kitchen vs. a bathroom. After all, agents must properly stop in the correct room to succeed. Furthermore, these representations should generalize beyond the small number of environments used in training, such as the 72 Gibson (Xia et al., 2018) scenes used in our experiments. However, this is a challenging generalization problem because the visual encoders (CNNs) are trained from scratch, and have never received semantic supervision of any kind (e.g., '*this is a kitchen*').

In this section, we compare the visual representations of different versions of our model using indoor scene classification as a proxy measure of generalization performance. Specifically, we report linear probing accuracy on the Places365 scene classification dataset (Zhou et al., 2017). Linear probes are commonly used in visual representation learning literature (e.g., He et al. 2020; Chen et al. 2020a) to measure the quality of learned representations. Places365 contains indoor and outdoor scene categories, however our

Figure 6. **Qualitative examples of a successful (left) and unsuccessful (right) navigation episode with our best agent (Table 1 Row 8).** [LEFT] The agent exits the bedroom and at $t = 15$ a low resolution view of a painting from the goal image becomes visible. Instead of entering the bathroom, the agent turns left and then successfully calls stop 0.65m from the goal at $t = 45$. [RIGHT] The agent enters the bedroom and incorrectly moves around the bed at $t = 60$, and then unsuccessfully stops 8.90m away from the goal. However, we qualitatively see some similarities between the agent's final observation at $t = 84$ and the goal image, such as the large windows.

agents were only trained in indoor environments. Thus, for this evaluation we use a subset of indoor scene classes that were identified by Du et al. (2021). Details in Appendix.

In Table 2, Row 4 is the performance of our general-purpose agent baseline using dense rewards and data augmentations applied independently over time. In Row 5, we observe that changing the augmentation strategy does not significantly change scene classification accuracy. However, adding a self-supervised auxiliary loss (Row 6) substantially improves accuracy, which is consistent with findings in 2D computer vision literature (e.g., Chen et al. 2020a).

In the sparse reward setting (Rows 7 and 8), we find that learning with sparse rewards improves scene classification performance. In particular, by comparing Row 6 vs. 8, we see that combining sparse rewards and SSL improves top-1 accuracy from 28.7% to 30.6% (+1.9%). These results indicate that training with sparse rewards can improve the visual representations learned by our agents.

5.6. Qualitative Example

In Figure 6, we present qualitative examples of our best agent searching for a goal image (bottom) in a new environment. In the successful episode on the left, the agent is initialized in a bedroom at $t = 0$. By $t = 15$, the agent has exited the bedroom and visual landmarks from the goal image, such as the painting, are visible in the distance at lower resolutions. The agent chooses to turn left instead of continuing straight into the bathroom. Then, it navigates towards the goal and successfully calls `STOP` 0.65m from the goal at $t = 45$. Due to minor deviations from the shortest-path (5.02m), the agents path length of 5.25m results in an SPL is 0.96. In the unsuccessful episode to the right, the agent incorrectly navigates around the bed and calls `STOP` 8.90m from the goal. We qualitatively find some similarities between the agent's final observation at $t = 84$ and the goal image, such as the large windows.

6. Discussion

In this work, we hypothesize dense reward shaping—as typically used for visual navigation—is problematic because it penalizes exploration. We argue that this may lead to learning spurious visual cues that support memorizing the training environment but do not generalize to new scenes. This leads us to explore a natural alternative: learning with sparse rewards. We address the challenges of learning from a sparse signal with self-supervised visual representation learning. Surprisingly, we find that SSL not only helps with learning from sparse rewards, but also allows sparse rewards to outperform the de facto standard—dense reward shaping.

Limitations. This work explores using strong and weak data augmentations within a reinforcement learning (RL) + self-supervised learning (SSL) framework. We choose specific augmentations based on prior work in visual navigation and visual representation learning, and we study effects under different RL reward structures. Alternative augmentation techniques may lead to different performance trade-offs. This work provides a foundation for exploring these alternatives in the context of learning from sparse rewards. Finally, while this work focused on a vision-only setting, it opens up a frontier for the study of a wider application of SSL + sparse rewards to other tasks such as object-goal navigation and vision-and-language navigation.

Acknowledgements

The Georgia Tech effort was supported in part by NSF, ONR YIP, and ARO PECASE. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Government, or any sponsor.

References

Anderson, P., Chang, A. X., Chaplot, D. S., Dosovitskiy, A., Gupta, S., Koltun, V., Kosecka, J., Malik, J., Mottaghi, R., Savva, M., and Zamir, A. R. On Evaluation of Embodied Navigation Agents. *arXiv preprint arXiv:1807.06757*, 2018a. 2, 6

Anderson, P., Wu, Q., Teney, D., Bruce, J., Johnson, M., Sünderhauf, N., Reid, I., Gould, S., and van den Hengel, A. Vision-and-language Navigation: Interpreting Visually-Grounded Navigation Instructions in Real Environments. In *CVPR*, 2018b. 2

Batra, D., Gokaslan, A., Kembhavi, A., Maksymets, O., Mottaghi, R., Savva, M., Toshev, A., and Wijmans, E. ObjectNav Revisited: On Evaluation of Embodied Agents Navigating to Objects. *arXiv preprint arXiv:2006.13171*, 2020. 2

Chaplot, D. S., Gandhi, D., Gupta, A., and Salakhutdinov, R. Object Goal Navigation using Goal-Oriented Semantic Exploration. In *NeurIPS*, 2020a. 2

Chaplot, D. S., Gandhi, D., Gupta, S., Gupta, A., and Salakhutdinov, R. Learning To Explore Using Active Neural SLAM. In *ICLR*, 2020b. 2, 5, 7

Chaplot, D. S., Salakhutdinov, R., Gupta, A., and Gupta, S. Neural Topological SLAM for Visual Navigation. In *CVPR*, 2020c. 2, 3, 5, 6, 7

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A Simple Framework for Contrastive Learning of Visual Representations. In *ICML*, 2020a. 2, 3, 5, 7, 8

Chen, X., Fan, H., Girshick, R., and He, K. Improved Baselines with Momentum Contrastive Learning. *arXiv preprint arXiv:2003.04297*, 2020b. 2, 5, 11

Du, Y., Gan, C., and Isola, P. Curious Representation Learning for Embodied Intelligence. In *ICCV*, 2021. 6, 8, 12

Hansen, N. and Wang, X. Generalization in Reinforcement Learning by Soft Data Augmentation. In *ICRA*, 2021. 3

Hansen, N., Jangir, R., Sun, Y., Alenyà, G., Abbeel, P., Efros, A. A., Pinto, L., and Wang, X. Self-supervised Policy Adaptation during Deployment. In *ICLR*, 2021. 3, 5

He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. In *CVPR*, 2016. 6

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum Contrast for Unsupervised Visual Representation Learning. In *CVPR*, 2020. 2, 3, 7

Hong, Y., Wu, Q., Qi, Y., Rodriguez-Opazo, C., and Gould, S. A Recurrent Vision-and-Language BERT for Navigation. In *CVPR*, 2021. 2

Kingma, D. P. and Ba, J. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*, 2014. 11

Kumar, A., Fu, Z., Pathak, D., and Malik, J. RMA: Rapid Motor Adaptation for Legged Robots. *arXiv preprint arXiv:2107.04034*, 2021. 1

Kwon, O., Kim, N., Choi, Y., Yoo, H., Park, J., and Oh, S. Visual Graph Memory with Unsupervised Representation for Visual Navigation. In *ICCV*, 2021. 1, 2

Laskin, M., Lee, K., Stooke, A., Pinto, L., Abbeel, P., and Srinivas, A. Reinforcement Learning with Augmented Data. In *NeurIPS*, 2020a. 4, 6

Laskin, M., Srinivas, A., and Abbeel, P. CURL: Contrastive Unsupervised Representations for Reinforcement Learning. In *ICML*, 2020b. 3

Loshchilov, I. and Hutter, F. SGDR: Stochastic Gradient Descent with Warm Restarts. *arXiv preprint arXiv:1608.03983*, 2016. 11

Maksymets, O., Cartillier, V., Gokaslan, A., Wijmans, E., Galuba, W., Lee, S., and Batra, D. THDA: Treasure Hunt Data Augmentation for Semantic Navigation. In *ICCV*, 2021. 1, 2, 4

Mezghani, L., Sukhbaatar, S., Lavril, T., Maksymets, O., Batra, D., Bojanowski, P., and Alahari, K. Memory-Augmented Reinforcement Learning for Image-Goal Navigation. *arXiv preprint arXiv:2101.05181*, 2021. 1, 2, 3, 4, 5, 6, 7, 11

Misra, I. and van der Maaten, L. Self-Supervised Learning of Pretext-Invariant Representations. In *CVPR*, 2020. 3

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *NeurIPS*, 2019. 4, 6

Ramakrishnan, S. K., Gokaslan, A., Wijmans, E., Maksymets, O., Clegg, A., Turner, J., Undersander, E., Galuba, W., Westbury, A., Chang, A. X., et al. Habitat-Matterport 3D Dataset (HM3D): 1000 Large-scale 3D Environments for Embodied AI. *arXiv preprint arXiv:2109.08238*, 2021. 4

Savva, M., Kadian, A., Maksymets, O., Zhao, Y., Wijmans, E., Jain, B., Straub, J., Liu, J., Koltun, V., Malik, J., Parikh, D., and Batra, D. Habitat: A Platform for Embodied AI Research. In *ICCV*, 2019. 2, 4, 6

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-Dimensional Continuous Control using Generalized Advantage Estimation. *arXiv preprint arXiv:1506.02438*, 2015. 5, 6

Stooke, A., Lee, K., Abbeel, P., and Laskin, M. Decoupling Representation Learning from Reinforcement Learning. In *ICML*, 2021. 3

Szot, A., Clegg, A., Undersander, E., Wijmans, E., Zhao, Y., Turner, J., Maestre, N., Mukadam, M., Chaplot, D. S., Maksymets, O., et al. Habitat 2.0: Training Home Assistants to Rearrange their Habitat. *NeurIPS*, 2021. 1

Tan, H., Yu, L., and Bansal, M. Learning to Navigate Unseen Environments: Back Translation with Environmental Dropout. In *NAACL*, 2019. 2

van den Oord, A., Li, Y., and Vinyals, O. Representation Learning with Contrastive Predictive Coding. *arXiv preprint arXiv:1807.03748*, 2018. 5

Wijmans, E., Kadian, A., Morcos, A., Lee, S., Essa, I., Parikh, D., Savva, M., and Batra, D. DD-PPO: Learning Near-perfect Pointgoal Navigators from 2.5 Billion Frames. In *ICLR*, 2019. 4, 6

Xia, F., Zamir, A. R., He, Z., Sax, A., Malik, J., and Savarese, S. Gibson Env: Real-world Perception for Embodied Agents. In *CVPR*, 2018. 4, 6, 7

Ye, J., Batra, D., Das, A., and Wijmans, E. Auxiliary Tasks and Exploration Enable ObjectGoal Navigation. In *ICCV*, 2021. 2

Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., and Torralba, A. Places: A 10 Million Image Database for Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 7, 11, 12

Zhu, Y., Mottaghi, R., Kolve, E., Lim, J. J., Gupta, A. K., Fei-Fei, L., and Farhadi, A. Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning. *ICRA*, 2017. 1, 2

A. ImageNav Training Hyperparameters

In Table 3, we list all of the hyperparameter used in our experiments. We selected the RL parameters based on (Mezghani et al., 2021) and SSL parameters based (Chen et al., 2020b). We did not sweep hyperparameters to find optimal values for our experiments.

Table 3. ImageNav Hyperparameters

RL Parameters	
number of GPUs	32
number of environments per GPU	4
rollout length	64
PPO epochs	4
number of batches per epoch	2
optimizer	Adam
learning rate	1.25e-4
epsilon	1e-5
PPO clip	0.2
generalized advantage estimation (GAE)	True
normalized	False
γ	0.99
τ	0.95
value loss coefficient	0.5
entropy loss coefficient	0.01
max grad norm	0.5
DD-PPO sync fraction	0.6
SSL Parameters	
momentum	0.999
temperature	0.07

B. Weak Data Augmentation

In Section 4.3, we describe *weak augmentations* that we use for reinforcement learning, which consist of spatial and color jitter. For spatial jittering, we using the `RandomResizedCrop` class from `torchvision` with `scale=(0.8,1.0)`. For color jittering, we use the `ColorJitter` class from `torchvision` with all initialization parameters set to `0.2`.

C. Strong Data Augmentation

In Section 4.4, we describe *strong augmentations* that we use for reinforcement learning, which consist of strong random cropping, horizontal flipping, grayscaling, color jittering, and Gaussian blurring. For strong random cropping, we using the `RandomResizedCrop` class from `torchvision` with `scale=(0.2,1.0)`. For color jittering, we use the `ColorJitter` class from `torchvision` with initialization parameters set to `(0.4, 0.4, 0.4, 0.1)`, and we apply the transform with a probability of `0.8`. We randomly grayscale the images with a probability of `0.2`, and with a probability of `0.5` we Gaussian blur the image using the `GaussianBlur` class from `torchvision` with `kernel_size=11` and `sigma=(0.1, 2.0)`.

D. Scene Classification Protocol

In Section 5.5, we report linear probe accuracy on Places365 (Zhou et al., 2017). For these evaluations we freeze the visual encoder trained for image-goal navigation and train a linear layer for indoor scene classification. The subset of indoor scene classes used in our experiments is listed in Appendix E. We train all models for 60 epochs using Adam (Kingma & Ba, 2014) with a learning rate of 1e-3 and use cosine-annealing (Loshchilov & Hutter, 2016). We train with `RandomResizedCrop` and `RandomHorizontalFip` data augmentations from `torchvision`, and use early stopping based on top-1 validation accuracy.

E. Indoor Scene Classes

Places365 (Zhou et al., 2017) consists of indoor and outdoor scenes. However, our agents have only been trained indoors, so we use the following subset of 55 scene classes selected by Du et al. (2021):

"classroom", "mansion", "patio", "airport_terminal", "beauty_salon", "closet", "dorm_room", "home_office", "bedroom", "engine_room", "hospital_room", "martial_arts_gym", "shed", "cockpit", "hotel-outdoor", "apartment_building-outdoor", "bookstore", "coffee_shop", "hotel_room", "shopfront", "conference_center", "shower", "conference_room", "motel", "fire_escape", "art_gallery", "art_studio", "corridor", "museum-indoor", "railroad_track", "inn-outdoor", "music_studio", "attic", "nursery", "auditorium", "residential_neighborhood", "cafeteria", "office", "restaurant", "waiting_room", "office_building", "restaurant_kitchen", "stage-indoor", "ballroom", "kitchen", "restaurant_patio", "staircase", "banquet_hall", "bar", "living_room", "swimming_pool-outdoor", "basement", "dining_room", "lobby", "locker_room."

This results in a training dataset size of 275,000 images and a validation set of 5,500 images.