FLAR: A Unified Prototype Framework for Few-sample Lifelong Active Recognition

Lei Fan, Peixi Xiong, Wei Wei and Ying Wu
Northwestern University, 2145 Sheridan Road, Evanston, IL, USA
{leifan, peixixiong2018, weiwei2022}@u.northwestern.edu, yingwu@northwestern.edu

Abstract

Intelligent agents with visual sensors are allowed to actively explore their observations for better recognition performance. This task is referred to as Active Recognition (AR). Currently, most methods toward AR are implemented under a fixed-category setting, which constrains their applicability in realistic scenarios that need to incrementally learn new classes without retraining from scratch. Further, collecting massive data for novel categories is expensive. To address this demand, in this paper, we propose a unified framework towards Few-sample Lifelong Active Recognition (FLAR), which aims at performing active recognition on progressively arising novel categories that only have few training samples. Three difficulties emerge with FLAR: the lifelong recognition policy learning, the knowledge preservation of old categories, and the lack of training samples. To this end, our approach integrates prototypes, a robust representation for limited training samples, into a reinforcement learning solution, which motivates the agent to move towards views resulting in more discriminative features. Catastrophic forgetting during lifelong learning is then alleviated with knowledge distillation. Extensive experiments across two datasets, respectively for object and scene recognition, demonstrate that even without large training samples, the proposed approach could learn to actively recognize novel categories in a class-incremental behavior.

1. Introduction

Visual recognition has been widely studied and achieved remarkable success in recent decades. In contrast to passive recognition from a still image, in robot learning scenarios, an intelligent agent is allowed to explore different viewpoints and is equipped with the capability to make decisions about what to observe. This problem is referred to as Active Recognition (AR), with two specific tasks illustrated in Figure 1(a).

A number of AR methods [4, 18, 12, 6, 23, 13, 7] have been proposed over the years with learning-based models.

Figure 1. FLAR is one challenging task that requires expanding dynamically to novel classes with few training samples. Meanwhile, the task setting fits the practical needs of robotic applications that need exploring previously unseen environments.
This problem is surprisingly under-explored in the literature but is imperative in realistic autonomous agent applications. In many scenarios, the agent trained in the backend to actively recognize fixed categories will be required to incrementally expand its AR ability for novel classes on the fly. We call this problem Lifelong Active Recognition (LAR), illustrated in Figure 1(b). Further, it is often costly to collect many training samples for novel categories, let alone the possibility that the samples of the novel category are scarce per se. This motivates us to study a new problem, Few-shot Lifelong Active Recognition (FLAR), that is both necessary and challenging. In Figure 1(c), the relationship of FLAR with closed problems is depicted.

Formally, FLAR raises three requirements to the agent. (1) The agent should be capable of making decisions to explore the most informative viewpoints based on the current stage so as to direct senses for a better understanding of surroundings. This fits into the realm of AR [32, 6, 4, 20, 41]. (2) The agent should adapt the power of exploration and recognition learned from old classes to new concepts while avoiding training from scratch. It is related to incremental learning [31, 30, 36]. (3) The agent should learn new concepts from limited training samples. It is connected with few-shot learning [37, 33, 15]. These requirements compose our FLAR problem, which delivers an intelligent agent that could incrementally learn to explore and recognize novel categories with only a few training samples.

Corresponding to these demands, three major challenges are posed by FLAR. (1) The previous AR methods typically learn a recognition policy from categories with massive training data. The constraint of few training samples for incremental categories will certainly impede the success of the policy training. (2) In the incremental learning setting, the agent is evaluated by the recognition performance of not only the categories on the fly but also old categories. Thus the catastrophic forgetting issue [30] needs to be tackled when new categories continuously emerge. (3) The risk of overfitting always exists for few-sample learning. Generalization from few samples needs to be fulfilled in our setting. In summary, FLAR is highly unconstrained with complex viewing conditions, growing recognition categories, and limited training samples.

In this paper, we propose a novel approach towards FLAR, a challenging but practical task that is under-explored. Although the challenges of FLAR scatter into different research fields, we address them in a unified framework. The main idea is that we hypothesize there exists a prototype in the embedding space to represent each category by averaging the aggregations of the budgeted exploratory observations for each sample. This facilitates flexible policy learning while simplifies knowledge preservation during class-incremental learning. Then for novel emerging categories, the agent is only required to take movements for the purpose of distinguishing the newly obtained features with prototypes of the trained categories. Note that the exemplars that best estimate prototypes are delicately selected and stored in the agent memory, which would be instilled during novel category learning.

Specifically, the insights of the proposed method aiming at the challenges of FLAR are three-fold: (1) The agent learns the active recognition policy based on a newly designed reward that favors a closer distance between aggregated features and the correct class prototype in the embedding space. (2) To handle the forgetting issue, only limited exemplars are stored in the agent memory in prioritized order. By reproducing consistent outputs for the exemplars utilizing the knowledge distillation mechanism, we incorporate the distribution of the old classes during learning novel concepts. (3) The prototype of each category, which serves as robust representations, potentially makes our approach adaptive to the few-training-sample challenge.

2. Related work

Active vision. Active vision has a long history in literature, which is pioneered by [2, 1, 8]. The common motivation behind these works is to bring intelligent control strategies to different visual tasks, i.e., the agent should actively obtain observations with its own purpose. Following this idea, active vision has been exploited in several lines, covering tasks like recognition [34, 4, 18, 26, 24, 38], navigation [39, 11, 5], localization [3] and scene completion [19, 29].

As a notable branch of active vision, prior AR approaches can be mainly identified into two groups based on whether explicitly measuring information gains between different views. For the method describing gains explicitly, [32] proposes a 3D saliency model to guide action selections. Others [6, 4] perform information gain maximization by representing the problem as a partially observable Markov decision process (POMDP). These methods tend to disambiguate among candidate labels with view-specific profits. On the other hand, there are approaches that undertake AR with deep reinforcement learning methods, in which policies are learned by gathering interaction experiences with the environment. For example, in [18], three modules, including control, single view recognition and evidence fusion, are composed in an end-to-end trainable system. The auxiliary task of predicting future observations helps to build correlations between views and movements. [12] considers establishing a consistent 3D latent model for each category by estimating depth and ego-motions from images. The policy aggregates the latent map recurrently to predict actions. In stark contrast to the FLAR task, most existing AR methods are performed with predetermined categories and do not support expanding to new classes. However, the agent exploration is inherently incremental: novel classes...
that could not be known in advance are demanded to be incorporated by the agent continuously.

**Lifelong learning.** Lifelong learning [27, 9, 30, 31, 36, 17, 35, 16, 21], also named continual learning, remains a long-standing challenge for machine learning since the catastrophic forgetting always occurs in non-stationary data distributions. Class-incremental learning requires progressively adding novel classes without restarting from scratch. A training strategy towards class-incremental learning criteria is firstly proposed in [30], where the knowledge distillation is utilized to maintain information from previous time points. [31] pays attention to incremental learning on the few-shot image classification by introducing meta-learning on an attention module. Other current works [36, 16] also intend to implement continual learning under various difficult conditions. Our approach, on the other hand, focuses on FLAR which comprises lifelong learning on the sequence-based decision and recognition process.

Recent adaptive agents methods [39, 25, 28, 14] adopt continual learning techniques for dynamic environments. The robot movement policy proposed in [28] could evolve with kinematic changes, e.g., to lose a wheel during exploration. These prospective works [39, 25] share similar motivation with our work, which is to address difficulties introduced by non-stationary environments. However, their method is unsuitable for FLAR since the knowledge preservation is not a precedent concern in these works.

**Few-shot learning.** There are plenty of works showing interest in few-shot image classification [33, 15, 37, 22, 42]. Their few-training-sample setting is closely related to the FLAR task. Various approaches have been delivered and could be roughly categorized into model-based, metric-based and optimization-based methods. In [33], they address few-shot image classification in the sense of metric learning by building a deep network as a function that could map the same-class inputs to the neighboring area in the embedding space. MAML [15] instead intends to learn a good parameter initialization of the base learner through gradients back-propagation. Our work shares a similar assumption with [33], i.e., the classifier should have a simple inductive bias to prevent overfitting with few samples. Following this assumption, our work towards FLAR introduces prototype representation during policy learning, which prompts an effective policy to obtain more informative views.

### 3. Method

For ease of presentation, we first define the setup and notations for FLAR. Then we describe three significant components of the proposed method and explain how their combination allows performing FLAR. An overview of our approach is demonstrated in Figure 2.

#### 3.1. Problem setup and notation

We describe our setting by applying it to an active object recognition scenario.

The active object recognition Agent is given an object instance \( x \) with an unknown label \( y \). A total of \( T \) timesteps is allowed for the Agent to predict the category of the object. At timestep \( t = 1, 2, \ldots, T - 1 \), the Agent could additionally select an action \( a \in \mathcal{A} \), e.g., to rotate up the object 30 degrees, where \( \mathcal{A} \) denotes the action space. As a result of taking movement, the visual sensor mounted on the Agent could get new observations of target instance \( x \). We assume the visual sensor remains at the same position while only rotating the object. To be more specific, the visual observation at time \( t \) is a 2D view as \( v_t = \mathcal{P}(x, p_t) \), where \( \mathcal{P}(\cdot, \cdot) \) is a projection function and \( p_t \) is the corresponding viewpoint. We evenly discretize the space of all viewpoints into a view-grid with the size of \( M \) azimuths \( \times N \) elevations. Then, each viewpoint can be designated to \( p_t = (m, n) \) with \( m \in \mathcal{M}, n \in \mathcal{N} \). The objective of Agent during a recognition episode is, therefore, three-fold, including making efficient exploration, aggregating observations among timesteps, and classification based on the fused information.

After introducing the recognition setting, we then characterize the detailed setup of incremental learning. As the Agent exploring in the environment, novel classes could occur at any time. A recognition task \( X \) indicates learning active recognition on specified categories. The Agent exploration can then be described as a class-incremental task stream \( X^{\text{base}}, X^1, X^2, \ldots, X^y, \ldots \). The \( X^{\text{base}} \) is the initial training set before the Agent exploration, which is from the \( C^{\text{base}} \) categories. Each following task, as training samples
of novel categories $C_{novel}$, is $X^y = \{x_1^y, \ldots, x_k^y\}, y \in C_{novel}$. Since the high expense of collecting training samples for newly discovered categories $y \in C_{novel}$, we limit the samples in $X^y$ by letting $|X^y| = k$, where $k$ is limited to 3, 5, 10 in our setting. For evaluation, the system is tested on the accuracy of its category prediction $\hat{y}$, and $\hat{y}$ belongs to seen candidate categories, i.e., $\hat{y} \in C_{seen}$ with $C_{seen} = C_{base} \cup C_{novel}$.

3.2. Prototype-guided active recognition

We comprehend AR as a procedure of achieving more discriminative features by reaching different views. Let us recall the basic motivation of AR, which is based on an observation that a single static image might not include enough information for classification, especially in an unconstrained environment. In other words, a static image might not be discriminative enough. The action selection of the agent can then be seen as a policy on the feature space, which should be rewarded if its representation becomes easier to differentiate from candidate categories.

In this part, we introduce our AR system in four steps. Firstly, we describe the representations that we want to learn during policy training. Then we introduce our recognition system architecture to achieve objective representations. The novel reward designed on our representation, as motivating attaining better views, is described. Finally, other losses for training our AR system are provided.

Prototype representation learning With a 3D object instance $x$, our recognition system could obtain a 2D view projection at time $t$, which is denoted as $v_t = P(x, p_t)$. The current view $v_t$ together with other proprioceptions, including both the relative position $p_{t-1,i}$ and the timestep $t$ itself, are regarded as the observations which is denoted as $X_t = h(v_t, p_{t-1,i}, t)$, where $h(\cdot)$ is a fuse operation.

We present our representation module as $f : \mathbb{R}^d \rightarrow \mathbb{R}^4$, a network to aggregate observations among timesteps. After actively taking $T - 1$ movements, the representation for object $x$ is $q_x = f(X_1, X_2, \ldots, X_T)$. Since training samples are limited, we hypothesize a prototype representation around which are aggregated features for the same class. Since the feature $q_x$ is extracted from multi-views and their relative poses, the correlation between views and viewpoints could be obtained during training, which, in other words, describes the object shape. Compared to the few-shot image classification method [33], the restrictive description ability of only one prototype could be alleviated to some extent by absorbing shape information.

For each category $y$, the prototype comes from a collected set $Q^y = \{q_1^y, q_2^y, \ldots\}$ as $\mu^y = \frac{1}{|Q^y|} \sum_{q \in Q^y} q$. Then, for an object instance $x$ with the representation $q_x$, its label is assigned as:

$$\hat{y} = \arg\min_y ||q_x - \mu^y||.$$  (1)

The label assignment is equal to $\hat{y} = \arg\max_y \mu^y q_x$, after we normalize representations. Therefore, we could consider the prototype of each category as a weight vector from our final linear layer, which is multiplied with $q_x$ for category probabilities. During training our AR system, we form the prototype representation learning with a loss term defined as:

$$\mathcal{L}_{category} = - \sum_i F_{softmax}(\hat{y}^i, y^i)$$ (2)

where $F_{softmax}$ is the softmax function, and the superscript $i$ denotes the corresponding training sample.

Active recognition system Our AR system is modeled on the architecture proposed in [18], which is mainly composed of three modules. The first module performs as a non-linear mapping function, which is previously defined as $f(\cdot)$. In our approach, we utilize a combination of a visual encoder and an LSTM network to recurrently fuse observations. The second module, i.e., the policy, could be treated as a Partially Observable Markov Decision Process (POMDP), whose pdf is defined as $\pi(a_t|X_{t-1}, \theta)$. $\theta$ is the parameter we want to obtain with policy gradients. This module is represented as a combination of linear layers in our approach, which predicts action distributions with the aggregated features. The third module is the classifier, i.e., a linear layer with the weight of prototypes for each category. At each timestep $t$, the proposed AR system selects an action with the highest probabilities. The classification is then applied to obtained temporally aggregated features.

Rewards for discrimination We design a novel reward to motivate the agent to choose views that result in more discriminative features. According to our classification in Equation 1, the discriminability ability between a feature and prototypes could be defined on their Euclidean distances. Intuitively, an increase of probabilities on the correct category represents the new feature becomes closer to the correct prototype among all candidates. We then define the reward $R(\hat{y}_t, \hat{y}_{t+1}) = 1$ as the growth of predicted probability of correct category, which promotes policy learning. Compared to a simple reward as $R(\hat{y}_t) = 1$ when the category prediction is correct, our proposed reward always focuses on achieving better views progressively.

The reward is used to train the policy via the reinforcement learning technique, i.e., REINFORCE, which could be back propagated to non-stochastic units. We define the loss for our policy learning as:

$$\mathcal{L}_{policy} = \sum_i \sum_{t=1}^{T} \log \pi(a_t_i|X_{t-1}, \theta) R(\hat{y}_t, \hat{y}_{t+1})^i.$$ (3)

Other Losses Two other terms, i.e., $\mathcal{L}_{entropy}$ and $\mathcal{L}_{forecast}$ are included during training our AR system. To promote more exploratory behavior of our agent and prevent policy collapse, the entropy loss $\mathcal{L}_{entropy}$ is calculated on the action distribution, which prefers selecting diversified actions.

15397
Another term $\mathcal{L}_{\text{forecast}}$ shares the same idea with [18] by introducing an auxiliary task of forecasting observations. Formally, we define this term as the following:

$$
\mathcal{L}_{\text{forecast}} = \sum_i \sum_{t=2}^{T} D(\hat{x}_i^t, x_i^t | x_i^{t-1}, a_{t-1}),
$$

where $D$ is a similarity measure as the cosine distance.

### 3.3. Lifelong learning on novel classes

The proposed approach to this point could only perform AR on fixed categories. Its classifier could not accommodate new classes during exploration. In this part, we present further details of our approach in handling lifelong learning.

**Agent memory** Catastrophic forgetting happens in our setting. One way to handle this challenge is to entangle weight in the classifier with the representation learning process. If not, the final output would change out of management [30]. The weight in our classifier is set to prototypes that change along with our representations learning. Therefore, the data distribution of previous classes should be introduced to the current training process to track prototype changes. We build a memory to store exemplars, i.e., object instances, that best describe the current category.

Only limited $m$ exemplars would be stored for each category to save memory space. The exemplar selection is conducted in a prioritized fashion [30]. An exemplar is selected if, by adding it to the memory, the average feature vector would best approximate the prototype overall training data. Such selection processes can be done one time for each category. The saved exemplar set for category $y$ is $M^y = \{x_1, x_2, \ldots, x_m\}$. In specific, the exemplars are saved in the form of view-grids for our approach since they are the direct visual inputs.

**Distillation loss** The knowledge of previous classes is maintained by encouraging reproducing the same output of the utilized datasets and our experimental setups. We first introduce the utilized datasets and our experimental setups. Then, we evaluate our approach in the class-incremental setting in Section 4.3. The comparison of our approach with other baselines is demonstrated in Section 4.4, which shows the effectiveness of our policy in AR. In Section 4.5, we conduct ablation studies on the training sample size and the saved exemplar size.

### 4. Experiments

To validate our approach to FLAR, we examine the performance on two challenging datasets. We first introduce the utilized datasets and our experimental setups. Then, we evaluate our approach in the class-incremental setting in Section 4.3. The comparison of our approach with other baselines is demonstrated in Section 4.4, which shows the effectiveness of our policy in AR. In Section 4.5, we conduct ablation studies on the training sample size and the saved exemplar size.

### 4.1. Datasets and experimental setups

We evaluate our approach on two widely used datasets for scene and object recognition, respectively.

**SUN360 dataset** The SUN360 scene dataset [40] contains spherical panoramas for 26 diverse scene categories. The dataset is split into 6174 training, 1013 validation, and 1805 testing examples. We test our agent on this dataset for active scene recognition. The field-of-view of our agent is limited to 60 degrees. The agent could rotate to move to novel objects. We implement recognition episodes on exemplars to achieve their classifier outputs before training new classes coming during exploration. In this part, we present further details of our approach in handling lifelong learning.

**Agent memory** Catastrophic forgetting happens in our setting. One way to handle this challenge is to entangle weight in the classifier with the representation learning process. If not, the final output would change out of management [30]. The weight in our classifier is set to prototypes that change along with our representations learning. Therefore, the data distribution of previous classes should be introduced to the current training process to track prototype changes. We build a memory to store exemplars, i.e., object instances, that best describe the current category.

Only limited $m$ exemplars would be stored for each category to save memory space. The exemplar selection is conducted in a prioritized fashion [30]. An exemplar is selected if, by adding it to the memory, the average feature vector would best approximate the prototype overall training data. Such selection processes can be done one time for each category. The saved exemplar set for category $y$ is $M^y = \{x_1, x_2, \ldots, x_m\}$. In specific, the exemplars are saved in the form of view-grids for our approach since they are the direct visual inputs.

**Distillation loss** The knowledge of previous classes is maintained by encouraging reproducing the same output of the utilized datasets and our experimental setups. We first introduce the utilized datasets and our experimental setups. Then, we evaluate our approach in the class-incremental setting in Section 4.3. The comparison of our approach with other baselines is demonstrated in Section 4.4, which shows the effectiveness of our policy in AR. In Section 4.5, we conduct ablation studies on the training sample size and the saved exemplar size.

### 4.1. Datasets and experimental setups

We evaluate our approach on two widely used datasets for scene and object recognition, respectively.

**SUN360 dataset** The SUN360 scene dataset [40] contains spherical panoramas for 26 diverse scene categories. The dataset is split into 6174 training, 1013 validation, and 1805 testing examples. We test our agent on this dataset for active scene recognition. The field-of-view of our agent is limited to 60 degrees. The agent could rotate to move to novel observations (shown in Figure 3). The agent needs an efficient recognition system could be successfully extended.

### 4.1. Datasets and experimental setups
Figure 3. The active scene recognition process of the proposed approach. Each row contains 3 steps, i.e., $t = 1, 3, 5$, from a recognition episode. The starting position is set to the same to show different trajectories on three samples. We mark the current view (green box) and the next movement grid (light yellow area). As shown in the first row, the proposed approach corrects its reasonable but wrong guesses within 5 steps. More visualizations of results from the ShapeNet dataset are included in the supplementary material.

$3 \times 5$ grid at each timestep.

We arrange the SUN360 dataset into a task stream to suit FLAR. We randomly select 16 categories as the initial task with all training samples, which forms the base category $C_{\text{base}}$. After the initial task, the agent is trained in a class-incremental way on the following 10 categories with limited samples. Each category represents a novel task. The performance is evaluated on test samples of the dataset, considering all classes that have already been trained. The amount $k$ of new-category training samples is limited, which is set to 5 if not specified. Moreover, only $m = 3$ exemplars are saved to memory without designation.

ShapeNet datasets Our experiment conducted on the ShapeNet dataset [10] considers the scenario that the agent could manipulate an object instance for recognition. The agent needs to predict its next best motion based on previous observations. Each training sample is a Computer-Aided Design (CAD) model. The view with the resolution of $32 \times 32$ is sampled from $M = 6$ camera elevations and $N = 12$ azimuths. For each step, the agent is able to move within 5 elevations-by-7 azimuths neighborhood of the current position. A total of 5 views could be achieved before giving final category predictions.

We randomly select 20 object categories from the ShapeNet dataset to conduct our experiments. Each category contains 35 samples for training, 10 samples for validation, and 10 samples for evaluation. Among 20 categories, we select 10 categories as the base category and form the other 10 categories into sequential tasks. The ShapeNet dataset is more challenging than the SUN360 scene dataset for two reasons. Firstly, it contains more view grids than in our SUN360 dataset setting, which, in other words, brings larger searching space for policy learning. Secondly, the synthetic 3D model might contain less texture information than a real object. And we also evaluate the performance of our approach with all seen categories.

4.2. Implementations

Our approach is implemented with PyTorch. The visual encoder of our approach is a simple 3-layer network with the ReLU activation. We utilize the recurrent neural network (LSTM) to aggregate temporal knowledge from observations. During trajectory gathering in reinforcement learning, we randomly provide the starting viewpoints. The exemplars are saved to the memory during training. Currently, we have not considered the limit of total memory size, which would be considered in our future work. We attach a classifier, i.e., a linear layer without bias, to the LSTM output at each step. The final classification result is reported as the average of class likelihoods of reached steps. We use current to denote the result based only on the current estimates and average as the final result.

4.3. Lifelong learning results

In this part of the experiment, we study the performance of the proposed approach under the full FLAR condition. Continual learning In this part, we intend to demonstrate the effectiveness of our approach in dealing with learning
novel categories. Since the proposed approach, to the best of our knowledge, is the first to address FLAR, we attempt to define the range of our performance by dramatically ease the task. We introduce the training setting that we name it Train From Scratch (TFS). The TFS could access all training data at the same time, which, in other words, is not constrained by both the forgetting or the few-sample challenges. Therefore, the closer of our performance with the result of TFS denotes the higher effectiveness of the proposed method, as we could achieve similar results by progressively learning novel classes with few samples.

Figure 4 shows the results on two datasets. The metric been utilized is classification accuracy. Note the TFS is re-trained with all data on each category setting while the proposed approach incrementally learns on novel classes without access to previous data. For each dataset, we display the result with timestep \( t = 1, 2, \ldots, 5 \). The performance of the proposed approach is the same as the TFS on the base category since no class-incremental learning is performed. One can see the effectiveness of our method in learning novel classes with few samples.

As expected, the overall performance of the proposed approach is better on the SUN360 dataset than on the ShapeNet dataset since the searching space for the ShapeNet dataset is obviously larger. Another finding revealed in Figure 4 is the performance arises with taking more steps. We will show the evaluation of our policy in Section 4.4 to explain the improvement is not only brought by obtaining more observations but also the policy.

An interesting observation from Figure 4 is the proposed approach occasionally exceeds the performance of the TFS on the ShapeNet dataset. It first positively suggests the prototype representation is effective in handling few-sample challenges. It also indicates large samples of a category might not bring direct benefits to the prototype representation since the prototype, i.e., the mean of features, would be distracted by several "hard" training samples.

**Imbalance of training data** We show the advantage of class-incremental learning from the aspect of data balance. We combine both samples from base categories and novel categories. Compared to TFS that can have sufficient training samples for novel categories, only limited \( k \) samples are provided in the Imbalance setting. Direct training under the Imbalance setting could be regarded as a long-tail/data-imbalance problem. In Figure 5, we demonstrate the Imbalance result with ours with incremental learning strategy. Our results at \( t = 5 \) steadily outperform the results from the Imbalance setting, which shows the effectiveness of our incremental learning strategy.
4.4. Comparison on AR

In this part, we would like to show the effectiveness of our policy on AR solely. We block our mechanism on class-incremental learning leaving only an AR agent for fixed categories. We first introduce the baselines.

**Single view:** The input is only a random starting view to our approach. No policy is needed in this method. We include this method in our comparison to show the performance of single view recognition.

**Random views:** This method shares a similar architecture with the proposed method, which replaces our policy module with a random action selection. The number of movements remains the same as ours.

**Largest step:** The policy is to take the movement that is the most distant to the current viewpoint. The assumption here is neighbor views usually share similar information.

**Look-Ahead** [18] is also based on a recurrent network architecture. The reward is defined as the current movement getting the correct prediction, which is different from our approach. It runs to the fixed steps as ours.

The comparisons on the SUN360 dataset [40] with all 26 categories are displayed in Table 1. The proposed approach could already outperform other methods on the basic AR task. The large improvements between our results with other passive baselines, including **Single view**, **Random views** and **Largest step**, denote the advantages of including effective policies during recognition. Our method is also better than [18] which is the result of two attributes. The first is our prototype representation learning which promotes obtaining structure consistencies between different object instances. The second reason is our novel reward that always motivates the agent to achieve more informative views. Note that the agent could intelligently stop taking further movements by redefining the action space.

4.5. Ablation studies

To provide further details of our approach, in this part, we perform our methods on the SUN360 dataset [40], in which we isolate its individual aspects.

**Sample size** First, we analyze the influence of the sample size on our method. We set the sample size \(k = 3, 5, 10\) while keeping the other parameters the same. The number of exemplars saved to memory is \(m = 3\). Our method is then trained in these three different settings. Figure 6(a) summarizes the results as the accuracy over all steps of the class-incremental learning. The results show that the number of samples actually contributes to the performance. In particular, by comparing the results of \(k = 5\) and \(k = 10\), one can see that the performance growth is not significant. We think it suggests that the proposed approach could obtain adequate prototype representations with 5 training samples. Another observation is the performance goes up after learning on class 26. The reason could be that our network achieves better prototypes of previous categories with saved exemplars during training on class 26, or class 26 introduces beneficial transferable knowledge.

**Exemplar size** We study how the number of exemplars influences our performances. The exemplar is chosen in a prioritized order after training on the current category. The result in Figure 6(b) are trained with \(m = 1, 3, 5\) with \(k = 5\). Note \(m = 1\) means only one exemplar that most approximate to the prototype is stored in the agent memory. The result demonstrates the effectiveness of the proposed approach when the memory size is extremely limited, which, in other words, validates our exemplar selection process.

5. Conclusions

In this paper, we propose a novel approach towards FLAR which incrementally learns active recognition on novel categories. Challenges, including few training samples and forgetting, are addressed with three major components. We derive the prototype as the representation for each category, which is robust in handling limited training samples. The novel designed reward motivates the agent to achieve more discriminative features by measuring distances in the embedding space. To alleviate catastrophic forgetting, the knowledge distillation with exemplars stored in the agent memory is applied during the policy training. The experimental results, along with ablation studies, show the effectiveness of proposed approach for the FLAR task. However, despite the promising result, FLAR is still a challenging task only at the beginning stage. We plan to study the influence of category relations to AR in our future work.

Acknowledgements

This work was supported in part by National Science Foundation grant IIS-1619078, IIS-1815561, and IIS-2007613.
References

[18] Dinesh Jayaraman and Kristen Grauman. Look-ahead before you leap: end-to-end active recognition by forecasting the effect of motion. In European Conference on Computer Vision, 2016. 1, 2, 4, 5, 8
through active observation completion. *Science Robotics*, 2019. 2, 5


[34] Stefano Soatto. Actionable information in vision. In *Machine Learning for Computer Vision*, 2013. 2


