REFERENCE-LIMITED COMPOSITIONAL LEARNING: A REALISTIC ASSESSMENT FOR HUMAN-LEVEL COMPOSITIONAL GENERALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

To narrow the considerable gap between artificial and human intelligence, we propose a new task, namely reference-limited compositional learning (RLCL), which reproduces three core challenges to mimic human perception: compositional learning, few-shot, and few referential compositions. Building upon the setting, we propose two benchmarks that consist of multiple datasets with diverse compositional labels, providing a suitable and realistic platform for systematically assessing progress on the task. Moreover, we extend popular few-shot and compositional learning approaches to serve as baselines, and also introduce a simple method that achieves better performance in recognizing unseen compositions. Extensive experiments demonstrate that existing solutions struggle with the challenges imposed by the RLCL task, revealing substantial research space for pursuing human-level compositional generalization ability.

1 INTRODUCTION

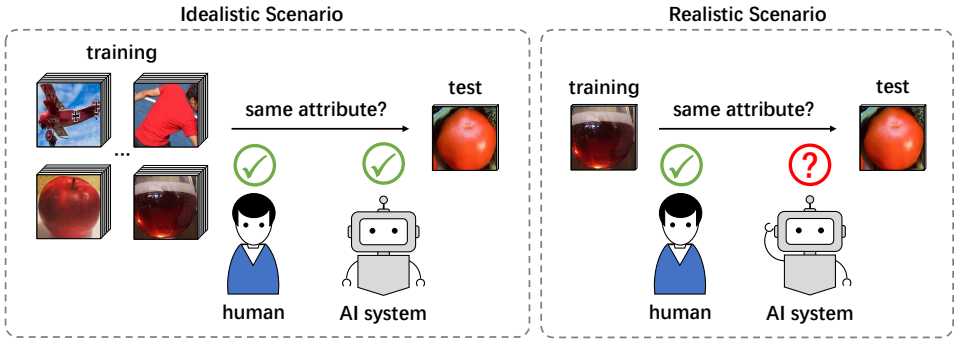

Figure 1: By learning from a large amount of seen compositions and samples as references, AI systems can extract the visual invariants to understand what is "red", and use the learned knowledge to recognize unseen compositional concepts. However, referential compositions and samples are usually insufficient when learning in realistic scenarios, making it more difficult to learn novel elements from limited compositions. While humans can rapidly generalize to unknown pairs of novel primitives in real-world environments, whether the artificial compositional learners can achieve such human-level compositional generalization ability is still a question. In this paper, we propose a realistic and untouched task to explore whether existing learning algorithms can still generalize well despite the limitations of few-shot and few referential compositions.

Although deep learning methods have made astounding progress in various domains of artificial intelligence (AI) research including computer vision (Goodfellow et al., 2016), they often take the assumption that training and test data are independent and identically distributed (i.i.d.). A growing number of investigations have demonstrated that these approaches do not generalize to out-of-distribution (o.o.d.) tests, *i.e.*, they fail to make correct predictions for examples that are even slightly out of the data distribution on which they are trained (Muandet et al., 2013; Motiian et al., 2017;

Arjovsky et al., 2019; Krueger et al., 2021). Such phenomenon is very common when deploying vision applications to the real world, and is essentially due to the fact that limited training data can only serve as an incomplete observation of an infinite number of situations in the world.

To endow AI systems with human-level generalization ability to quickly adapt to new environments, researchers began to seek inspiration from how humans learn and understand the world. Different from standard systems that are limited to a fixed set of categories at a time, humans generalize to a large, essentially "unbounded" concept space by reasoning in a compositional manner (Bahdanau et al., 2019; Vedantam et al., 2021). For example, based on familiarity with tomatoes and other red objects, people can recognize a red tomato when they first encounter it. Similarly, if one has cut a cake, it is easy to recognize the behavior of cutting a pizza after knowing what a pizza is. This method of identifying novel complex concepts by composing known components (which we call the "*primitives*" in this paper) is called compositional generalization, representing the essential ability of human intelligence to make "infinite use of finite means" (Chomsky, 1957; Humboldt, 1988).

While considerable attention has been devoted to natural language processing works that improve generalization performance on test data by equipping compositional skills (Lake & Baroni, 2018; Finegan-Dollak et al., 2018; Russin et al., 2020), visual perception models are also expected to more accurately identify compositional concepts, which could be significantly different from their semantic constituents (Sadeghi & Farhadi, 2011; Zhang et al., 2017). Further, prior studies on attribute-object compositions have begun to consider unseen compositions, that is, they learn the compositionality of objects and their states from sufficient training samples and are tasked with generalizing to unseen combinations of these primitives (Misra et al., 2017; Nagarajan & Grauman, 2018; Purushwalkam et al., 2019; Naeem et al., 2021). While these efforts have contributed to a more comprehensive perception of the world, we argue that the existing setup seems idealistic and inappropriate to simulate natural human learning. Firstly, humans have an inherent ability to learn the compositionality of complex concepts with only a few examples and transfer the learned knowledge to different situations. However, the **few-shot** problem would lead to severer generalization issues in AI systems as the empirical risk is far from being a good approximation for expected risk (Wang et al., 2020). Although an increasing number of models have tried to alleviate potential overfitting (Snell et al., 2017; Finn et al., 2017; Chen et al., 2019), they still treat every class as an independent entity and require referential data for any novel concept. Hence we would like to investigate whether compositional learning can be performed with a restricted sample size, in other words, whether few-shot learners can generalize to unseen label compositions. A more neglected point is, unlike existing methods that have to refer to a large number of combinations with the same primitive to extract semantic invariants from them, humans can discover potential primitives from few combinations, or even only one, based on prior knowledge. This contributes to the adaptation of humans to the long-tailed distribution of various compositional concepts in the real world, where exist a few common primitives and many more composition-scarce primitives, making collecting all possible scenarios for each primitive in advance expensive and time-consuming. Therefore, **few referential compositions** should also be a natural constraint for human-level compositional learning.

In this paper, we propose a novel task, **reference-limited compositional learning** (RLCL), that approximates the naturalistic learning environment that humans and artificial agents encounter. The term "reference-limited" is used to indicate that when the model performs compositional learning, the combinations that can be utilized as references are limited in terms of both the number of categories and the number of labeled samples. Therefore, RLCL requires the learner to incorporate appropriate priors into learning, so that it can disentangle features of primitives without superfluous references. As existing benchmarks fail to provide suitable conditions for systematic comparisons on RLCL task, we provide two benchmarks that consist of multiple datasets of natural images attached with sufficient attribute-object and action-object compositional labels, supporting us to sample realistic episodes to simulate partially observable worlds. We also develop a simple refined-ProtoNet that utilizes class descriptions to adaptively separate features of the specific primitive. Compared to few-shot and compositional learning methods extended to our RLCL task, our method obtains significant gains on recognizing unseen compositions. However, the mediocre results indicate challenges introduced by RLCL remain to be further addressed. By shedding light on the limitations of existing settings and approaches, we hope to spur future work to develop human-level compositional generalization ability for intelligent systems. We summarize our contributions as follows:

1. We introduce a new task denoted as reference-limited compositional learning (RLCL), where the model is required to learn the primitives and generalize to recognize unseen compositions given only

a few samples of limited compositions that contains these primitives. This offers a more realistic and challenging environment for training and evaluating compositional learners.

2. We establish two datasets with diverse compositional labels and well-designed data splits, providing the required platform for systematically assessing progress on the task. Moreover, we adapt classic few-shot and compositional learning methods and also propose refined-ProtoNet together as baselines to accelerate future studies.

3. We conduct extensive experiments and analyses to explore the impact of constraints introduced by the RLCL task. While refined-ProtoNet consistently achieves superior performance on recognizing unseen compositions, experimental results show that the performance of these approaches is hindered, leaving substantial room for developing human-level compositional generalization ability.

2 REFERENCE-LIMITED COMPOSITIONAL LEARNING

2.1 PROBLEM FORMULATION

The ultimate goal of the RLCL task is to recognize unseen visual pair compositions, whose primitives have only appeared in limited seen compositions containing only a few samples. In this paper, we follow the FSL setting to use the sampled episodes as a simulation of independent test environments, which refer to the data for learning as *support* and the data for inference as *query*. In addition, we apply an **open world** setting that while all compositions contained in the *support classes* are *seen* ones, the *query classes* include not only *unseen compositions*, but also *seen compositions*. At the same time, we impose no constraint on the test time search space. Allowing predictions to come from all possible pairs in the current episode, the setting is closer to the scenarios that are likely to arise in real-world deployments, and thus leads to a more comprehensive study on achieving a balanced and promising performance of both seen and unseen compositions.

More formally, we consider the visual recognition setting where each image x is associated with a complex concept c that is a pair composition of two primitives p^1 and p^2, *i.e.*, $c = (p^1, p^2)$. For example, p^1 can represent a state like "cooked" or an action like "cut", while p^2 can refer to an object such as "chicken" or "pizza". When testing, the model are evaluated on episodes that are sampled from a set of *novel* data $\mathcal{D}_n = \{(x_n^{(i)}, c_n^{(i)})\}$ with label space $\mathcal{C}_n \subset \mathcal{P}_n^1 \times \mathcal{P}_n^2 = \{(p_n^1, p_n^2) | p_n^1 \in \mathcal{P}_n^1, p_n^2 \in \mathcal{P}_n^2\}$. $\mathcal{C}_n$ denotes the novel composition set, and $\mathcal{P}_n^1$, $\mathcal{P}_n^2$ are the two corresponding novel primitive sets, each with N^p primitive categories. Each episode contains a *support* set $\mathcal{S} = \{(x_s^{(i)}, c_s^{(i)}) | i = 1, 2, \ldots, N_s^c \times K_s^c\}$ that consists of N_s^c support classes with K_s^c labeled samples per class, and a *query* set $\mathcal{Q} = \{(x_q^{(i)}, c_q^{(i)}) | i = 1, 2, \ldots, N_q^c \times K_q^c\}$ that consists of N_q^c query classes with K_q^c samples per class. The query classes not only contain N_s^c *seen* compositions that are all in the support classes, but also comprise $(N_q^c - N_s^c)$ *unseen* compositions that do not overlap with seen compositions. However, seen and unseen compositions in the same episode share the same two primitive sets sampled from $\mathcal{C}_n^1$ and $\mathcal{C}_n^2$, providing the possibility for unseen compositions to be recognized. Following the open world setting, the prediction space of the model contains N^{p2} compositions including seen, unseen and unfeasible ones. And the goal of the model is to correctly predict the compositional labels of samples in $\mathcal{Q}$ with the access to $\mathcal{S}$.

To extract the prior knowledge for learning to rapidly separate primitive features from images, in the training phase, the model possesses the access to a set of *base* data $\mathcal{D}_b = \{(x_b^{(i)}, c_b^{(i)})\}$ with label space $\mathcal{C}_b \subset \mathcal{P}_b^1 \times \mathcal{P}_b^2$. Note that the base and novel primitive sets do not overlap, *i.e.*, $\mathcal{P}_b^1 \cap \mathcal{P}_n^1 = \emptyset$ and $\mathcal{P}_b^2 \cap \mathcal{P}_n^2 = \emptyset$, and thus $\mathcal{C}_b \cap \mathcal{C}_n = \emptyset$ also holds. We would like to mention that RLCL does not require a specific procedure for learning from the training data. Keeping the spirit of matching training and test conditions, some baselines follow the episodic training paradigm, meaning that they are trained on episodes sampled using the same algorithm as used for test episodes. And the non-episodic baselines are trained using all the labeled samples in $\mathcal{D}_b$.

2.2 EPISODE SAMPLING

In this section, we outline the sampling strategy that creates more realistic episodes for the RLCL task. In each experiment, the value of N^p is fixed for all sampled episodes in the same phase. However, we allow episodes to have a different number of seen and unseen compositions, *i.e.*, the

values of N_s^c and N_q^c may vary from episode to episode. As N_q^c actually corresponds to the number of all potential compositions that can be obtained by pairing all primitives in the episode and also exist in the dataset, N_s^c is randomly sampled within a certain range of $[N^p + 1, N_q^c - 1]$, making the episode closer to the reality. The maximum value of this range guarantees that there exist unseen compositions in the episode, and the minimum value implies that each primitive has the opportunity to appear in more than one seen composition without being bound to another primitive all the time.

Concretely, for each episode, we first randomly sample N^p seen compositions to obtain two primitive sets $\mathcal{P}_n^1$, $\mathcal{P}_n^2$ without duplicate primitives. Then, we check if $\mathcal{P}_n^1$ and $\mathcal{P}_n^2$ can be paired to get enough existing compositions for being divided into seen and unseen ones. An episode that can achieve the required number of composition pairs will be regarded as a valid episode, and the remaining compositions will be randomly assigned to seen and unseen groups on the premise of satisfying the restriction of N_s^c. Therefore, we have N_s^c seen compositions for the support classes and N_q^c compositions including seen and unseen ones for the query classes. Next, we randomly sample K_s^c support samples for each seen composition and K_q^c samples for each composition in the query classes. Note that for seen compositions that exist in both support and query classes, the samples assigned to the two sets are not duplicated. Corresponding pseudocode can be found in Appendix D.

2.3 BENCHMARK DATASETS

As none of the current few-shot learning benchmarks provides the real-world compositional reasoning challenges that we would like to study, we create two datasets RLCL-ATTR and RLCL-ACT, aiming to offer an environment for progress measurement and controlled analysis of the RLCL task. Each dataset is comprised of multiple different existing datasets to ensure the presence of a sufficient amount of data and composition pairs. This allows us to create a sufficient number of episodes with diversity to simulate possible new environments during the test phase, ultimately ensuring the validity of the evaluation. Therefore, according to its data sources, each of RLCL-ATTR and RLCL-ACT can be naturally divided into three parts for different phases, and the primitives, compositions and data contained in each part do not overlap. Also, to drive research in realistic recognition tasks covering a variety of compositional concepts, we consider attribute-object and action-object, the two most frequent composition types in the natural data, for the datasets. In other words, our benchmark requires evaluated methods for a compositional learning of three types of primitives: actions, attributes and objects, making itself more comprehensive and challenging. We give a general overview of the two datasets and describe how we cleaned the existing data to get them. More details of these datasets are provided in Appendix C.

RLCL-ATTR: This dataset consists of 99,771 images attached with 1,768 **attribute-object** pair labels. Among them, 51,928 images of 1,076 compositions in the training phase are obtained from C-GQA (Naeem et al., 2021), 29,922 images of 136 compositions in the validation phase are obtained from UT-Zap50K (Yu & Grauman, 2014), and 17,921 images of 556 compositions in the test phase are obtained from MIT-States (Isola et al., 2015).

RLCL-ACT: This dataset contains 30,420 images with 574 **action-object** pair labels, where 20,604 images of 214 compositions in the training phase are from HICO (Chao et al., 2015), 1,207 images of 22 compositions in the validation phase are from Visual Genome (Krishna et al., 2017), and 8,609 images of 338 compositions in the test phase are from imSitu (Yatskar et al., 2016).

To ensure that the two datasets we obtained meet the needs of the RLCL task, we filtered the data in all splits from the perspective of labels. Specifically, (1) compositions with fewer than 10 samples were screened out to ensure that enough support and query samples can be simultaneously sampled from the composition without duplicates, (2) for primitives that appeared in multiple splits, we kept them in at most one split, (3) size-related attribute primitives that cannot be accurately depicted in the images, such as “small”, “large”, “long” and “short”, were also filtered out from our datasets. Specially, images from the Visual Genome dataset are densely annotated with numerous attributes and objects, lacking a description of the focus of the content. Therefore, we kept the attribute-object compositional label of the largest bounding box in each image, which most likely corresponds to the main content, and removed other annotations.

2.4 CHALLENGES OF THE RLCL TASK

We here summarize the three major challenges in the RLCL setting and discuss how they impair the generalization performance of the models.

Compositional learning: To recognize new compositions that were never observed during training, the model is required to shift the target of prediction from compositions to the primitive categories. Thus, the primitive features should be disentangled from the samples and used for both training and inference of the classifiers. However, as the representations of primitives are likely to be inherently entangled, separating which primitive each part of the features belongs to can be challenging. Meanwhile, due to the principle of contextuality (Misra et al., 2017), the same primitive will be displayed differently according to its matching context. As a result, images associated with the same primitive may exhibit large intra-class variation, making it more difficult to learn generalized classifiers.

Few-shot: As recognizing unseen compositions can be viewed as an extreme case of distribution-shift inference (Atzmon et al., 2020), the trained classifiers may overfit to the feature distributions encoded only from the seen compositions. And a series of FSL studies have proved that such phenomenon can be exacerbated by limiting the number of labeled samples. In our RLCL setup, taking the RLCL-ATTR dataset as an example, each seen composition has only N_s^c (no more than 5 in the experiments) samples to learn in the episode. For comparison, there is an average of 28.43 samples per seen composition in the MIT-State datasets (Isola et al., 2015) (the source of test split of RLCL-ATTR) in the CZSL task. Moreover, a greater disparity arises when considering the amount of referential data available for primitives. Each primitive has only 6.39 support samples on average in each episode of RLCL-ATTR, while possessing 343.72 training samples on average in MIT-States.

Few referential compositions: As we have pointed out, observing a primitive from different contexts can help to extract the visual invariants and disambiguate its semantic properties. Therefore, a large number and variety of referential compositions promote the ability to segregate primitive features from the images, contributing to the rising performance of existing compositional learning methods. While each primitive occupies 12.16 seen compositions on average in MIT-States, RLCL places a limit on the number of seen compositions that the model can refer to. Setting N^p to 5 as default, each primitive appears in only 1.28 seen compositions on average in each test episode of RLCL-ATTR. This limitation further increases the difficulty of feature separation and the risk of overfitting, making RLCL a unique and hard-to-solve problem.

2.5 EVALUATION METRICS

In the RLCL task, we focus on how well the learned model recognizes both unseen and seen composition pairs, consistent with the adopted open world setting. To this end, we introduce three evaluation metrics: (1) **Unseen accuracy** (UA): The average of the accuracy computed on query samples from unseen compositions on all test episodes. (2) **Seen accuracy** (SA): The average of the accuracy computed on query samples from seen compositions on all test episodes. (3) **Harmonic mean** (HM): A metric that quantifies the overall performance of both seen and unseen accuracy based on the results of all test episodes, defined as: $\text{HM} = 2\left(\text{SA} * \text{UA}\right) / \left(\text{SA} + \text{UA}\right)$. Since the value of UA is usually much smaller than SA, the level of HM is often dominated by the level of UA, which is also consistent with our greater interest in how well the compositional learners recognize unseen compositions. Thus, in most cases, HM can be used as the most representative metric to participate in the comparison. In addition, we also record the statistics of each type of primitive computed from the query samples of unseen and seen compositions.

3 METHODS

3.1 BASELINES

To accelerate future studies on RLCL, we here implement and extend several popular few-shot and compositional learning methods as baselines. Specifically, MAML (Finn et al., 2017), ANIL (Raghu et al., 2020), ProtoNet (Snell et al., 2017), RelationNet (Sung et al., 2018), Baseline (Chen et al., 2019) and Baseline++ (Chen et al., 2019) are included due to their convenience of expansion, proven performance, and representativeness among various FSL methods. For all selected FSL methods, as they are designed to treat each class as an independent entity without considering pair labels, we

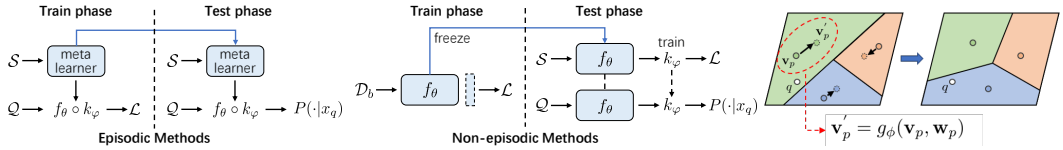

Figure 2: **Left:** Episodic baselines on RLCL. **Middle:** Non-episodic baselines on RLCL. We use $\mathcal{L}$ to denote the loss, f_θ to denote the backbone, and k_φ to denote two independent primitive classifiers or a joint compatibility classifier for compositions, depending on the specific method. **Right:** our refined-ProtoNet utilizes the class description $\mathbf{w}_p$ to adaptively separate features related to the corresponding primitive p, which can be regarded as a rectification to the initial prototype $\mathbf{v}$.

retain their training process, however, adapt them to have two parallel classifiers while sharing a backbone. Each classifier is the same as the original but is responsible for a single type of primitive rather than the entire label. And the loss also becomes the sum of the losses calculated by the two classifiers. We have noticed that many recent FSL efforts claim that they can outperform the state-of-the-arts on widely-used benchmarks. However, we point out that many methods rely on elaborate arithmetic and structural designs, and establishing a parallel classification branch for each primitive type would result in doubling the number of parameters and computation time. Moreover, we also experimented with some of them in advance, and results showed that they struggled to achieve desirable performance with highly entangled image representations. Thus, they cannot be adapted to our RLCL task by simply adding a parallel classifier, and we leave it to follow-up studies on how to make these methods available for compositional learning with non-trivial extensions. And for CZSL methods, we first consider VisualProd (Nagarajan & Grauman, 2018) that is commonly used in existing works and does not require auxiliary side information. To further evaluate the highest performance that existing CZSL methods can achieve on our RLCL task, we additionally select and extend two state-of-the-art methods, SymNet (Li et al., 2020) and CGE (Naeem et al., 2021), which are provided with word embeddings of labels as side information. For baselines adapted from CZSL methods, we use all base data $\mathcal{D}$ to train the feature backbone which is then fixed, and support data in each test episode to train the episode-specific classifiers, imitating the strategy that applies a pre-trained feature extractor and trains other parameters with data from downstream tasks. More details of these baseline models are given in Appendix F.

3.2 REFINED-PROTONET

The results in Table 1 show that ProtoNet performs relatively well in the implemented baselines, especially in terms of SA. Nevertheless, the poor UA reveals that computing the mean of same-primitive features is not an effective substitute for feature separation, since both primitive prototypes and query features involved in the prototypical classification may be mixed with features of another primitive type. This has little impact on the SA under the limitation of few referential compositions, as each seen composition is likely to be the only one that contains the two primitives. However, it leads to non-negligible degradation of UA. To devise a more competitive baseline for the RLCL task, we propose that an adaptive feature selection mechanism should be designed to remedy the shortcoming of ProtoNet. Such a mechanism is expected to enhance the features of corresponding primitives according to class descriptions and suppress the irrelevant features, thus serving as a separation. Considering that each channel of a feature map can be regarded as a feature detector (Zeiler & Fergus, 2014), and each dimension of the learned prototypes actually corresponds to each channel of the original feature map, it is natural that this adaptive feature selection should act on channels.

Therefore, according to the above discussion, we develop a simple yet effective approach namely refined-ProtoNet, which learns to utilize class descriptions to selectively emphasize features related to the corresponding primitives based on the extended ProtoNet. Formally, for a specific primitive p, we can obtain the refined prototype $\mathbf{v}_p^{'}$ that can then be used for prototype classification:

$$\mathbf{v}_p^{'} = g_\phi(\mathbf{v}_p, \mathbf{w}_p),\ \mathbf{v}_p = \frac{1}{|\mathcal{S}_p|} \sum_{(x_s^{(i)}, c_s^{(i)}) \in \mathcal{S}_p} \mathrm{GAP}(\mathbf{F}_s^{(i)}), \tag{1}$$

where GAP denotes global average pooling operation. $\mathbf{F} = f_\theta(x) \in \mathbb{R}^{C\times H\times W}$ is the feature map output by the backbone f_θ, and $\mathcal{S}_p$ denotes the support samples of the primitive p. g_ϕ is the feature selection function with parameters ϕ for the corresponding classification branch, $\mathbf{w}_p$ denotes the class description of p, and $\mathbf{v}_p$ denotes the prototype of p, *i.e.*, an average of features that are extracted from support samples that contain p in their compositional labels. Depending on whether side information can be obtained, we propose two options for designing g_ϕ, $\mathbf{w}_p$ and $\mathbf{v}_p$. The experimental results in Section 4 show that refined-ProtoNet with either option improves significantly in identifying unseen compositions, making it a competitive baseline that must be considered for defeat in future studies.

Side information as $\mathbf{w}_p$. Assuming that the side information is available, as is often the case in zero-shot learning, we can use pre-trained word embeddings of the p as the class descriptions $\mathbf{w}_p \in \mathbb{R}^{D_\mathbf{w}}$. While making a soft selection of the features of the prototypes in the channel dimension, we also perform an adaptive mixing with the mapped class descriptions to obtain the final refined prototypes, thus allowing human prior knowledge to compensate for the sample scarcity problem caused by the few-shot challenge. Formally, we have

$$\mathbf{v}_p^{'} = g_\phi(\mathbf{v}_p, \mathbf{w}_p) = \frac{1}{1+\exp(-\sigma(h_1(\mathbf{w}_p^{'})))} \otimes \mathbf{v}_p + \frac{\exp(-\sigma(h_1(\mathbf{w}_p^{'})))}{1+\exp(-\sigma(h_1(\mathbf{w}_p^{'})))} \otimes \mathbf{w}_p^{'}, \tag{2}$$

$$\mathbf{w}_p^{'} = \delta(h_0(\mathbf{w}_p)), \tag{3}$$

where σ denotes the Sigmoid activation function, δ denotes the ReLU activation function, and $\otimes$ denotes element-wise multiplication. $h_0(\cdot), h_1(\cdot)$ are two fully-connected feed-forward networks, each of which consists of two linear transformations with a ReLU activation in between. h_0: $\mathbb{R}^{D_\mathbf{w}} \rightarrow \mathbb{R}^C$ maps $\mathbf{w}_p$ to the space where $\mathbf{v}_p$ is located, and h_1: $\mathbb{R}^C \rightarrow \mathbb{R}^C$ is used to generate a weight for each dimension with side information. The probability of predicting x_q as p can be expressed as

$$P(p|x_q) = \text{softmax}(-d_e(\text{GAP}(\mathbf{F}_q), \mathbf{v}_p^{'})), \tag{4}$$

where d_e denotes the squared euclidean distance.

Prototypical features as $\mathbf{w}_p$. However, when coming to a new environment, AI systems learning to recognize new primitives and compositions do not necessarily have human prior knowledge of these new categories. When the side information is not available, we propose that prototypical features can also be used as class descriptions. Inspired by Woo et al. (2018), we use both global average pooling and max pooling to generate channel-wise statistics. Therefore, we have:

$$\mathbf{v}_p^{'} = g_\phi(\mathbf{v}_p, \mathbf{w}_p) = g_\phi(\mathbf{v}_p, \mathbf{F}_p) = \sigma(h(\text{GMP}(\mathbf{F}_p)) + h(\text{GAP}(\mathbf{F}_p))) \otimes \mathbf{v}_p, \tag{5}$$

$$\mathbf{F}_p = \frac{1}{|\mathcal{S}_p|} \sum_{(x_s^{(i)}, c_s^{(i)}) \in \mathcal{S}_p} \mathbf{F}_s^{(i)}, \tag{6}$$

where $h(\cdot)$ denotes two fully-connected layers around the ReLU non-linearity, and GMP denotes global max pooling operation. As the only input to this feature selection is the visual features themselves, the same selection can be done for features of query samples in each classification branch, isolating the features corresponding to the primitive type of the current branch:

$$P(p|x_q) = \text{softmax}(-d_e(g_\phi(\text{GAP}(\mathbf{F}_q), \mathbf{F}_q), \mathbf{v}_p^{'})), \tag{7}$$

Table 1: Results (%) with 95% confidence intervals under default parameter settings. Our refined-ProtoNet achieves the best results on both UA and HM in both two datasets.

Method	RLCL-ATTR			RLCL-ACT		
	UA	SA	HM	UA	SA	HM
MAML	2.87±0.50	35.38±1.69	5.32±0.87	1.99±0.27	29.56±2.03	3.72±0.45
ANIL	1.53±0.17	27.86±0.52	2.91±0.31	1.68±0.51	24.95±2.95	3.14±0.88
ProtoNet	2.23±0.99	**45.97**±1.83	4.25±1.80	2.75±0.90	**44.89**±1.31	5.17±1.60
RelationNet	1.82±0.39	33.55±1.26	3.46±0.69	1.19±0.46	30.28±5.55	2.28±0.84
Baseline	1.59±0.35	35.67±0.63	3.04±0.65	1.23±0.20	30.75±1.23	2.36±0.37
Baseline++	0.76±0.10	20.42±0.33	1.46±0.19	0.28±0.04	18.69±1.17	0.55±0.08
VisualProd	0.55±0.45	15.83±0.60	1.07±0.83	0.18±0.14	16.19±0.32	0.35±0.28
refined-ProtoNet	**4.36**±0.84	42.03±1.58	**7.89**±1.36	**3.60**±0.89	36.92±1.57	**6.55**±1.45
Learning with Side Information						
SymNet	1.96±0.95	18.47±0.68	3.54±1.54	2.96±0.35	17.12±0.53	5.04±0.53
CGE	4.10±1.09	17.03±0.13	6.61±1.43	2.73±0.78	19.12±0.65	4.78±1.17
refined-ProtoNet	**6.05**±1.64	39.31±0.73	**10.48**±2.44	**6.18**±1.20	32.04±4.78	**10.35**±1.65

4 EXPERIMENTS

Experimental Setup. For a fair comparison, the basic experiments are conducted with a four-layer convolution backbone (Conv-4) as in (Chen et al., 2019) for all implemented methods. If not specified, K_s^c, K_q^c, and N^p are all set by default to 5 while N_s^c and N_q^c are dynamic and randomly sampled in each episode. All methods take in images resized to 84×84 as input. In the training stage, we apply standard data augmentation including random crop, left-right flip, and color jitter. All methods are trained using the Adam (Kingma & Ba, 2015) optimizer with an initial learning rate 10^{-3} and a L2 penalty of 5×10^{-4}. We train 60,000 episodes for episodic methods and 600 epochs for non-episodic methods. For methods that require training parameters in test episodes, we use the entire support set to train for 100 iterations. For methods using side information, we initialize the word embeddings with pre-trained 300-dimensional word2vec (Mikolov et al., 2013) vectors. And the best model is selected with the HM performance on the validation set. The reported average results with 95% confidence intervals are obtained over 3 random experiments. Datasets and code implemented in PyTorch (Paszke et al., 2019) will be released upon acceptance.

Our experiments aim to answer the following research questions:

How well do the implemented methods perform on RLCL? To first provide an overview of how the implemented methods perform, Table 1 presents the UA, SA, and HM of all methods on two datasets. As can be observed, all methods are much more successful in identifying seen compositions than unseen compositions. Since identifying the former essentially does not require compositional generalization ability, the poor UA reveals that existing methods fail to learn in a compositional manner under the constraint of few-shot and few referential compositions. On top of this conclusion, giving the credit to our adaptive feature selection mechanism, refined-ProtoNet shows a significant advantage in identify unseen compositions and thus achieves the highest HM.

How much the few-shot challenge inhibits the compositional learning? Starting from 5, we gradually decrease K_s^c to 3 and 1. As illustrated in Figure 3 (left), the top-performing methods are declining as the number of samples available for reference decreases, and the majority of methods perform worst when $K_s^c = 1$. We also report the harmonic mean of different types of primitive in seen and unseen compositions in Appendix H. Similarly, in most cases, the overall performance of the model in predicting the primitives gets worse as K_s^c decreases. And our refined-ProtoNet continuously maintain outstanding performance together with ProtoNet. Another observation worth noting is that in RLCL-ATTR, a majority of methods predict objects with higher accuracy than attributes, presumably because recognizing the attributes is more difficult in this dataset. However, different methods show different tendencies in the accuracy of predicting actions and objects in RLCL-ACT, indicating that recognizing action-object pairs is a more complicated task that requires carefully designed solutions.

How much the few referential compositions challenge inhibits the compositional learning? To get an answer to this question, we conduct an experiment to fix the total number of compositions

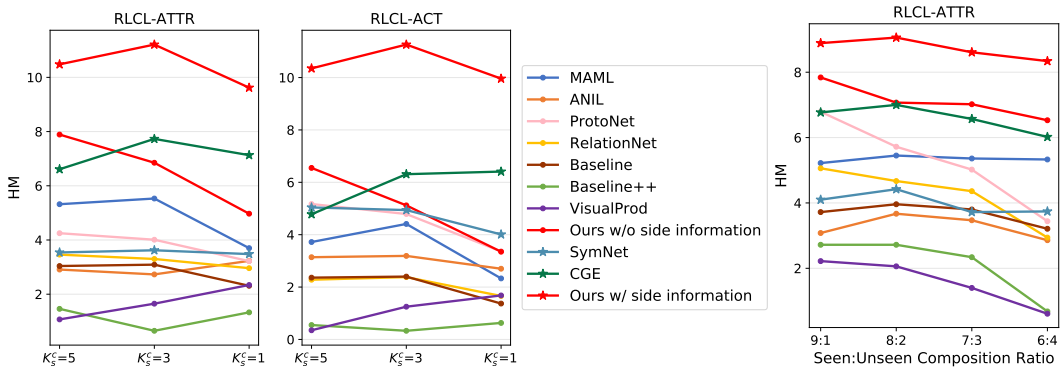

Figure 3: **Left:** HM (%) of different K_s^c on the two datasets. **Right:** HM (%) of different seen-unseen composition ratios on RLCL-ATTR. As both the few-shot and few referential compositions challenges become more extreme, the overall performance of these methods generally weakens. Methods using side information take a pentagram (★) as the marker, and the methods not using side information take a dot (●) as the marker. Our refined-ProtoNet is marked with a red line.

in each test episode and constantly adjust the seen:unseen composition ratio. The results are shown in Figure 3 (right), and we note that the experiment is conducted only on RLCL-ATTR as its test split supports for generating a sufficient number of episodes with various ratios. Unsurprisingly, the fewer seen compositions are available, the worse the overall performance of all methods is. This demonstrates that when the composition pairs available for reference become scarce, it is more difficult for the model to correctly separate the primitive features from the composition features. Apart from this conclusion, our refined-ProtoNet achieves the best HM on all ratios.

Why does refined-ProtoNet have advantages and disadvantages over ProtoNet in different metrics? As can be observed in previous experimental results, refined-ProtoNet consistently outperforms ProtoNet on both UA and HM but has worse performance on SA. Moreover, according to Table 4 and 5, the performance of the two methods in recognizing primitives is essentially the same on RLCL-ATTR, while on RLCL-ACT ProtoNet is even slightly better. To analyze the reasons for the appearance of such phenomenon, we compare the errors that refined-ProtoNet makes to those of ProtoNet. With refined-ProtoNet, about 70% of unseen compositions (U) are confused for seen compositions (S), and about 25% of unseen compositions are confused for incorrect unseen pairs in RLCL-ATTR. This yields an rate of $\frac{U \to S}{U \to U} = \frac{70\%}{25\%} = 2.8$. And when side information is available, this rate is further reduced to $\frac{69\%}{25\%} = 2.76$. For comparison, this rate of ProtoNet is $\frac{85\%}{12\%} \approx 7.08$. In RLCL-ACT, the rate of refined-ProtoNet is $\frac{U \to S}{U \to U} = \frac{71\%}{25\%} = 2.84$ when not using side information and $\frac{58\%}{36\%} \approx 1.61$ when using side information , while the one of ProtoNet is $\frac{86\%}{11\%} \approx 7.82$. As such rates of ProtoNet are much more unbalanced than those of refined-ProtoNet, ProtoNet is more likely to be overfitted to the seen compositions. As a result, it sacrifices performance in recognizing unseen compositions, and ultimately results in worse HM. On the contrary, refined-ProtoNet can better combine the decomposed primitive features to recognize various compositions, which is consistent with our expectations for compositional learners.

5 Conclusion

In this paper, we introduce reference-limited compositional learning (RLCL), a novel and non-trivial task that mimics the naturalistic learning environment for compositional learners. To present a rich playground to drive research on the task, we build two datasets that consist of natural images attached with various compositional labels, which are commonly used to describe the world. Furthermore, baselines adapted from popular few-shot and compositional learning algorithms are also provided. Combining the challenges of compositional learning, few-shot, and few referential compositions, our RLCL task has been proven by extensive experimental results that cannot be properly solved by existing methods. We hope our work can facilitate and calibrate the development of compositional learning systems that can be deployed in the real world.

REFERENCES

Martín Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Yuval Atzmon, Felix Kreuk, Uri Shalit, and Gal Chechik. A causal view of compositional zero-shot recognition. In *Proceedings of the Advances in Neural Information Processing Systems*, 2020.

Dzmitry Bahdanau, Shikhar Murty, Michael Noukhovitch, Thien Huu Nguyen, Harm de Vries, and Aaron C. Courville. Systematic generalization: What is required and can it be learned? In *Proceedings of the International Conference on Learning Representations*, 2019.

Yu-Wei Chao, Zhan Wang, Yugeng He, Jiaxuan Wang, and Jia Deng. HICO: A benchmark for recognizing human-object interactions in images. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1017–1025, 2015.

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. In *Proceedings of the International Conference on Learning Representations*, 2019.

Noam Chomsky. Logical structures in language. *American Documentation*, 8(4):284, 1957.

Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir R. Radev. Improving text-to-sql evaluation methodology. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 351–360, 2018.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the Proceedings of the International Conference on Machine Learning*, pp. 1126–1135, 2017.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *Proceedings of the International Conference on Learning Representations*, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Siteng Huang, Min Zhang, Yachen Kang, and Donglin Wang. Attributes-guided and pure-visual attention alignment for few-shot recognition. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pp. 7840–7847, 2021.

Drew A. Hudson and Christopher D. Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6700–6709, 2019.

Wilhelm von Humboldt. The diversity of human language-structure and its influence on the mental development of mankind, 1988.

Phillip Isola, Joseph J. Lim, and Edward H. Adelson. Discovering states and transformations in image collections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1383–1391, 2015.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross B. Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1988–1997, 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*, 2015.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73, 2017.

David Krueger, Ethan Caballero, Jörn-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Rémi Le Priol, and Aaron C. Courville. Out-of-distribution generalization via risk extrapolation (rex). In *Proceedings of the Proceedings of the International Conference on Machine Learning*, pp. 5815–5826, 2021.

Brenden M. Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the Proceedings of the International Conference on Machine Learning*, pp. 2879–2888, 2018.

Yonglu Li, Yue Xu, Xiaohan Mao, and Cewu Lu. Symmetry and group in attribute-object compositions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11313–11322, 2020.

Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 3111–3119, 2013.

Ishan Misra, Abhinav Gupta, and Martial Hebert. From red wine to red tomato: Composition with context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1160–1169, 2017.

Saeid Motiian, Marco Piccirilli, Donald A. Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5716–5726, 2017.

Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *Proceedings of the Proceedings of the International Conference on Machine Learning*, pp. 10–18, 2013.

Muhammad Ferjad Naeem, Yongqin Xian, Federico Tombari, and Zeynep Akata. Learning graph embeddings for compositional zero-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

Tushar Nagarajan and Kristen Grauman. Attributes as operators: Factorizing unseen attribute-object compositions. In *Proceedings of the European Conference on Computer Vision*, pp. 172–190, 2018.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1532–1543, 2014.

Senthil Purushwalkam, Maximilian Nickel, Abhinav Gupta, and Marc'Aurelio Ranzato. Task-driven modular networks for zero-shot compositional learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3592–3601, 2019.

Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of MAML. In *Proceedings of the International Conference on Learning Representations*, 2020.

Laura Ruis, Jacob Andreas, Marco Baroni, Diane Bouchacourt, and Brenden M. Lake. A benchmark for systematic generalization in grounded language understanding. In *Proceedings of the Advances in Neural Information Processing Systems*, 2020.

Jacob Russin, Jason Jo, Randall C. O'Reilly, and Yoshua Bengio. Compositional generalization by factorizing alignment and translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pp. 313–327, 2020.

Mohammad Amin Sadeghi and Ali Farhadi. Recognition using visual phrases. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1745–1752, 2011.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Proceedings of the Advances in Neural Information Processing Systems*, pp. 4077–4087, 2017.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1199–1208, 2018.

Pavel Tokmakov, Yu-Xiong Wang, and Martial Hebert. Learning compositional representations for few-shot recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 6372–6381, 2019.

Ramakrishna Vedantam, Arthur Szlam, Maximilian Nickel, Ari Morcos, and Brenden M. Lake. CURI: A benchmark for productive concept learning under uncertainty. In *Proceedings of the Proceedings of the International Conference on Machine Learning*, pp. 10519–10529, 2021.

Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys (CSUR)*, 53(3):63:1–63:34, 2020.

Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. CBAM: convolutional block attention module. In *Proceedings of the European Conference on Computer Vision*, pp. 3–19, 2018.

Mark Yatskar, Luke S. Zettlemoyer, and Ali Farhadi. Situation recognition: Visual semantic role labeling for image understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5534–5542, 2016.

Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set functions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8805–8814, 2020.

Aron Yu and Kristen Grauman. Fine-grained visual comparisons with local learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 192–199, 2014.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Proceedings of the European Conference on Computer Vision*, pp. 818–833, 2014.

Hanwang Zhang, Zawlin Kyaw, Shih-Fu Chang, and Tat-Seng Chua. Visual translation embedding network for visual relation detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3107–3115, 2017.

Yixiong Zou, Shanghang Zhang, Ke Chen, Yonghong Tian, Yaowei Wang, and José M. F. Moura. Compositional few-shot recognition with primitive discovery and enhancing. In *Proceedings of the ACM International Conference on Multimedia*, pp. 156–164, 2020.

A Related Work

A.1 Compositional Learning

As the concept of compositionality was introduced very early in the philosophy of language, most prior works (Lake & Baroni, 2018; Finegan-Dollak et al., 2018; Russin et al., 2020) focused on encouraging neural networks to achieve human-like compositional generalization in understanding natural language. While the compositionality of natural language is explicitly reflected in applying known syntactic and grammatical rules to novel words, cues in visual environments are more implicit and uncertain. Therefore, in addition to the feature hierarchy applied by convolutional networks, how to better achieve the reuse of features corresponding to semantic elements for vision systems remains to be explored. Some prior efforts (Johnson et al., 2017; Hudson & Manning, 2019; Ruis et al., 2020) focused on grounded language understanding task as the testbed, while the compositional generalization ability of visual perception models was not investigated independently. A more relevant topic to our work is compositional zero-shot learning (CZSL) (Misra et al., 2017; Nagarajan & Grauman, 2018; Li et al., 2020), which aims to recognize unseen attribute-object compositions at test time while each constituent exists in training samples. Using side information that describes novel composition pairs, *e.g.*, word embeddings, attribute annotations, or text descriptions, some notable methods utilize modular networks (Purushwalkam et al., 2019) or graph convolutional networks (Naeem et al., 2021) to learn a joint compatibility function between the image, the attribute, and the object. However, RLCL focuses more broadly on compositional learning of various types of primitives based on the scarcity of referential compositions and samples. Experimental results show that even the state-of-the-art CZSL methods also struggle with this challenge and are surpassed by simple extensions on classical prototypical networks.

A.2 Few-Shot Learning

Few-shot learning (FSL) requires learning new tasks with few labeled examples. Recent FSL advances can be roughly categorized into the following three groups: (1) *metric-based* methods (Snell et al., 2017; Sung et al., 2018; Ye et al., 2020) learn a generalizable embedding model to transform all samples into a common metric space, where simple classifiers can be executed directly. (2) *initialization-based* methods (Finn et al., 2017; Raghu et al., 2020) learn a good set of initial parameters for the whole model or part of it, so that the model can quickly adapt to novel classes in a small number of gradient update steps. (3) *pretraining-based* methods (Chen et al., 2019) train a feature extractor with all the training data, and fix it during the meta-test phase whilst learning new classifiers for novel classes.

Recently, several FSL works have aimed to improve the generalization performance with compositional representations. Tokmakov et al. (2019) propose two forms of regularizations to learn an image representation that is decomposable into parts by leveraging category-level attribute annotations. Zou et al. (2020) use the self-supervision from object split orders to discover part-related primitives, which are then composed to enhance novel classes. Huang et al. (2021) utilize an attributes-guided attention mechanism to learn a more informative image representation as a combination of local semantic features. However, limited by the traditional FSL setting on which they are based, these approaches only consider feature compositionality and have not explored how to generalize to new label compositions.

B Setting Comparison

Table 2: Comparison of RLCL to other settings. Notation "-" indicates the corresponding challenge is not considered by the FSL setting due to the non-compositional prediction targets.

	Few-Shot	Compositional	Few Referential Compositions
FSL	✓	✗	-
CZSL	✗	✓	✗
RLCL (Ours)	✓	✓	✓

In this paper, we present a novel and non-trivial task, namely reference-limited compositional learning (RLCL). To clarify the differences between RLCL and existing settings, we include Table 2 to list the major challenges faced by these settings. In contrast to *few-shot learning* (FSL), RLCL requires to recognize compositional concepts instead of independent entities, explicitly evaluating the compositional generalization ability of visual models. Also, while there exists *compositional zero-shot learning* (CZSL) that studies to recognize unseen compositions, it does not limit either the number of referential compositions or the number of labeled samples per composition, making it far from real-world deployment scenarios of intelligent systems.

C Dataset Details

In this section, we give a more detailed introduction for our proposed RLCL-ATTR and RLCL-ACT datasets.

C.1 Basic Statistics

Table 3: Basic statistics of our proposed datasets.

Dataset	RLCL-ATTR	RLCL-ACT
Composition type	attribute-object	action-object
Total number of c	1,768	574
Number of c in train / val / test	1,076 / 136 / 556	214 / 22 / 338
Total number of p^1	190	185
Number of p^1 in train / val / test	105 / 33 / 52	52 / 10 / 123
Total number of p^2	488	154
Number of p^2 in train / val / test	281 / 12 / 195	59 / 11 / 84
Total number of samples	99,771	30,420
Number of samples in train / val / test	51,928 / 29,922 / 17,921	20,604 / 1,207 / 8,609

Basic statistics of the datasets are included in Table 3.

C.2 Dataset Sources

RLCL-ATTR and RLCL-ACT are formed of data originating from different real-world image datasets. We here introduce the datasets we use in each split after further filtering. Note that the statistics presented in the following are only used to describe the original datasets.

For RLCL-ATTR, the **training split** is from Compositional GQA (C-GQA) (Naeem et al., 2021) (Figure 4(a)), a recently proposed dataset built on top of Stanford GQA dataset (Hudson & Manning, 2019). The dataset contains 9,378 compositional labels across 38k images. The **validation split** is from UT-Zappos50K (Yu & Grauman, 2014) (Figure 4(b)), a large shoe dataset with 50,025 catalog images of shoe type-material pairs from Zappos.com. The dataset is created in the context of an online shopping task, where users care specifically about fine-grained visual differences. And the **test split** is from MIT-States (Isola et al., 2015) (Figure 4(c)), a dataset that contains 63,440 images depicting 245 natural objects in 115 different states, forming 1,262 possible composition pairs in total.

For RLCL-ACT, the **training split** is from Humans Interacting with Common Objects (HICO) (Chao et al., 2015) (Figure 4(d)), a dataset that consists of a total of 47,774 images, covering 600 categories of sense-based human-object interactions over 117 common actions performed on 80 common objects. The **validation split** is from Visual Genome (Krishna et al., 2017) (Figure 4(e)), a dataset that collects dense annotations of objects, attributes, and relationships within each images. The dataset contains 108,077 images where each image has an average of 35 objects, 26 attributes, and 21 pairwise relationships between objects. And the **test split** is from imSitu (Yatskar et al., 2016) (Figure 4(f)), a large-scale situation recognition dataset that contains over 125,000 images depicting 200,000 distinct situations. Each situation includes one of 500 possible activities and objects from 11,000 options.

(a) C-GQA (b) UT-Zappos50K (c) MIT-States

(d) HICO (e) Visual Genome (f) imSitu

Figure 4: Examples taken from each dataset forming RLCL-ATTR and RLCL-ACT.

D Episode Sampling Strategy

To better describe the episode sampling algorithm used in RLCL, we illustrate the pseudocode in the Algorithm 1 that has been implemented in our code.

E Experimental Setup

All methods take in images resized to 84×84 as input. In the training stage, we apply standard data augmentation including random crop, left-right flip, and color jitter. All methods are trained using the Adam (Kingma & Ba, 2015) optimizer with an initial learning rate 10^{-3} and a L2 penalty of 5×10^{-4}. We train 60,000 episodes for episodic methods and 600 epochs for non-episodic methods. For methods that require training parameters in test episodes, we use the entire support set to train for 100 iterations. For methods using side information, we initialize the word embeddings with pre-trained 300-dimensional word2vec (Mikolov et al., 2013) vectors. And the best model is selected with the HM performance on the validation set. Datasets and code implemented in PyTorch (Paszke et al., 2019) will be released upon acceptance.

F Baseline Implementation Details

As we have discussed in Section A, existing FSL and CZSL approaches can not attempt the proposed benchmarks without extensions. In this section, we introduce all baselines and how we adapt them for comparison. Note that RedWine (Misra et al., 2017) was also taken into account and implemented at first, however, eventually removed due to its poor performance especially when $K_s^c = 1$.

MAML. Short for Model-Agnostic Meta-Learning (Finn et al., 2017), MAML is a fairly general optimization algorithm. To achieve a good generalization across a variety of episodes, MAML aims to find the optimal initial parameters for the model, so that it can be rapidly and efficiently fine-tuned to the new episode. Therefore, in the training episodes, the fine-tuned model parameters are

Algorithm 1: Episode Sampling in RLCL

Input: $\mathcal{D}$ with label space $\mathcal{C}$ according to the requested class split of the given dataset, N^p, K_s^c, K_q^c

Output: the sampled episode $\{\mathcal{S}, \mathcal{Q}\}$

Step 1. Sample primitive sets $\mathcal{P}^1$, $\mathcal{P}^2$

$\mathcal{P}^1 = \{\}$, $\mathcal{P}^2 = \{\}$, seen compositions $\mathcal{C}_{\text{seen}} = \{\}$;

while $|\mathcal{C}_{seen}| < N^p$ **do**

  Randomly sample a composition $c = (p^1, p^2)$ from $\mathcal{C}$;

  if *(p^1 not in $\mathcal{P}^1$) and (p^2 not in $\mathcal{P}^2$)* **then**

    Add c into $\mathcal{C}_{\text{seen}}$, p^1 into $\mathcal{P}^1$, and p^2 into $\mathcal{P}^2$;

Step 2. Sample the support set $\mathcal{S}$

$\mathcal{S} = \{\}$, candidate compositions $\mathcal{C}_{\text{candidate}} = \{\}$, unseen compositions $\mathcal{C}_{\text{unseen}} = \{\}$;

for $c \in \mathcal{P}^1 \times \mathcal{P}^2$ **do**

  if *(c in $\mathcal{C}$) and (c not in $\mathcal{C}_{seen}$)* **then**

    Add c into $\mathcal{C}_{\text{candidate}}$;

if $|\mathcal{C}_{candidate}| < 2$ **then** `// Seen and unseen compositions are insufficient`

  Jump back to **Step 1**;

Randomly assign the first two compositions in $\mathcal{C}_{\text{candidate}}$ to each of $\mathcal{C}_{\text{seen}}$ and $\mathcal{C}_{\text{unseen}}$, and the remaining ones in $\mathcal{C}_{\text{candidate}}$ are randomly assigned to either $\mathcal{C}_{\text{seen}}$ or $\mathcal{C}_{\text{unseen}}$ each time;

for $c \in C_{seen}$ **do**

  Randomly sample K_s^c samples of c into $\mathcal{S}$;

Step 3. Sample the query set $\mathcal{Q}$

$\mathcal{Q} = \{\}$;

for $c \in C_{seen}$ **do**

  Randomly sample K_q^c samples of c from those do not overlap with $\mathcal{S}$ into $\mathcal{Q}$;

for $c \in C_{unseen}$ **do**

  Randomly sample K_q^c samples of c into $\mathcal{Q}$;

first obtained by performing a few gradient descent steps on the support set, and then the meta-optimization is performed over the initial model parameters by backpropagating the second-order gradients computed with the query set. For the RLCL task, we employ two independent linear layers followed by a softmax function on the top of the backbone network, each of which is responsible for predicting one type of primitives. The step size of gradient descent steps is set to 0.4.

ANIL. Short for Almost No Inner Loop (Raghu et al., 2020), ANIL is a significant simplification to MAML that removes the inner loop updates for all but the head (final layer) of a neural network during training and testing. Reported results show that ANIL performs identically to MAML on standard few-shot classification benchmarks and offers computational benefits over MAML. Similar to MAML, for the RLCL task, each type of primitive owns an independent linear classifier with inner loop updates, while the backbone network without the inner loop is shared.

ProtoNet. Short for Prototypical Networks (Snell et al., 2017), ProtoNet averages the representations of support samples from the same class as the prototype, and then classifies each query sample as the class whose prototype is "nearest" to it under Euclidean distance. For the RLCL task, we construct a prototype for each primitive by aggregating the samples of compositions that contain this primitive. Therefore, for each query sample, we perform two nearest-neighbor searches and each of them predicts a primitive.

RelationNet. Short for Relation Networks (Sung et al., 2018), RelationNet applies a learnable non-linear relation module instead of a fixed nearest-neighbor or linear classifier to evaluate the relationship of the query image and category embeddings. The relation module consists of two convolutional blocks and two fully-connected layers. Each convolutional block is a 3×3 convolution with 64 filters followed by batch normalization, ReLU non-linearity, and 2×2 max-pooling. The non-linearities after the two fully-connected layers are ReLU and Sigmoid, respectively.

Baseline. As a non-episodic approach, Baseline (Chen et al., 2019) uses all the labeled samples in the training stage to train the model, which consists of a backbone network as the feature extractor and a linear classifier followed by a softmax function. When testing, the parameters of the backbone network are fixed, and a new classifier is trained with the support set of each new episode. For the RLCL task, we apply two linear classifiers to separately predict the primitives for each input image. Following the original implementation, the batch size is set to 16 in the training stage, and each new classifier is trained using a SGD optimizer for 100 iterations with a batch size of 4, an initial learning rate 10^{-2} and a L2 penalty of 10^{-3}.

Baseline++. Very similar to the Baseline, Baseline++ (Chen et al., 2019) uses the cosine distance between the input feature and the learned weight vectors representing each class, aiming to reduce intra-class variations. The weights of this distance-based classifier can be interpreted as prototypes for each class. Here we also set up two separate classifiers as the extension for RLCL. Adjusting the original value range [-1,1] to be more appropriate for the subsequent softmax layer, we follow the original implementation to multiply the cosine similarity by a class-wise learnable scalar. The other hyperparameters are the same as those used in the Baseline.

VisualProd. This is a common discriminatively-trained CZSL baseline (Misra et al., 2017; Nagarajan & Grauman, 2018), which uses two independent classifiers over image features to predict the two primitives. The probability of a composition pair is simply the product of the probability of each primitive: $P(c) = P(p^1)P(p^2)$. We implement each classifier with a feedforward network with two fully-connected layers, and ReLU non-linearity is used between the layers.

SymNet. Inspired by group theory, SymNet (Li et al., 2020) implements symmetry and the group axioms including closure, associativity, identity element, invertibility element as the learning objectives. And attribute classification is accomplished based on a Relative Moving Distance recognition method, which utilizes the attribute change instead of the attribute pattern itself. The trade-off hyperparameters of the symmetry, axiom, triplet, p^1 and p^2 classification loss are set to 0.5, 0.01, 0.03, 1.0 and 0.1.

CGE. Short for Compositional Graph Embedding (Naeem et al., 2021), CGE learns a globally consistent joint embedding space between image features and seen and unseen compositions from a graph, where nodes are connected if a dependency exists in form of a compositional label, *e.g.*, cute, dog and cute dog. For each test episode, we construct a new graph with all primitives and possible compositions in the episode. The classification loss of support samples is backpropagated through the seen compositional nodes to parameters of the graph convolutional network and the image embedding function.

G Additional Experimental Results

In this section, our additional experiments aim to answer the following additional questions:

Is the model selection strategy for RLCL appropriate? Including choosing hyperparameters, training checkpoints and architecture variants, effective model selection is crucial for obtaining promising performance. However, when evaluating the generalization ability of the models with distribution shifts, model selection is not as straightforward as in supervised learning due to the lack of access to a validation set identically distributed to the test data (Gulrajani & Lopez-Paz, 2021). Following the common selection strategy used in FSL, we create a held-out validation set and randomly sample a certain number of validation episodes, on which the model maximizing the validation metric (HM in RLCL) is chosen. As the strategy assumes that validation and test episodes are drawn from the same meta-distribution over episodes, one concern is that, unlike the standard FSL, the data sources of validation set and test set are different in our proposed datasets, and therefore there exist distribution shifts that are not just caused by class differences. This may result in a severer deviation between the actual situation and the assumption, thus affecting the effectiveness of the validation.

To confirm that our validation set can play the role of model selection on the RLCL task, we record the metrics (classification loss, UA, SA, and HM) on the same number of validation and test episodes simultaneously during the training of refined-ProtoNet without side information. As illustrated in Figure 5, we apply a simple moving average smoothing with a weight of 0.9 to better show the trend of each metric. It can be observed that despite the differences in the specific values, the trends of

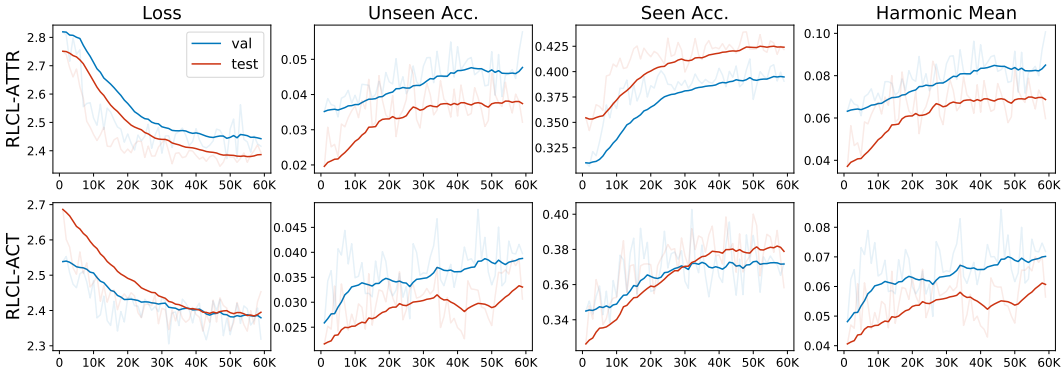

Figure 5: Metrics of refined-ProtoNet on validation and test episodes as the training progresses. We increased the transparency of the original lines to highlight the smoothed version.

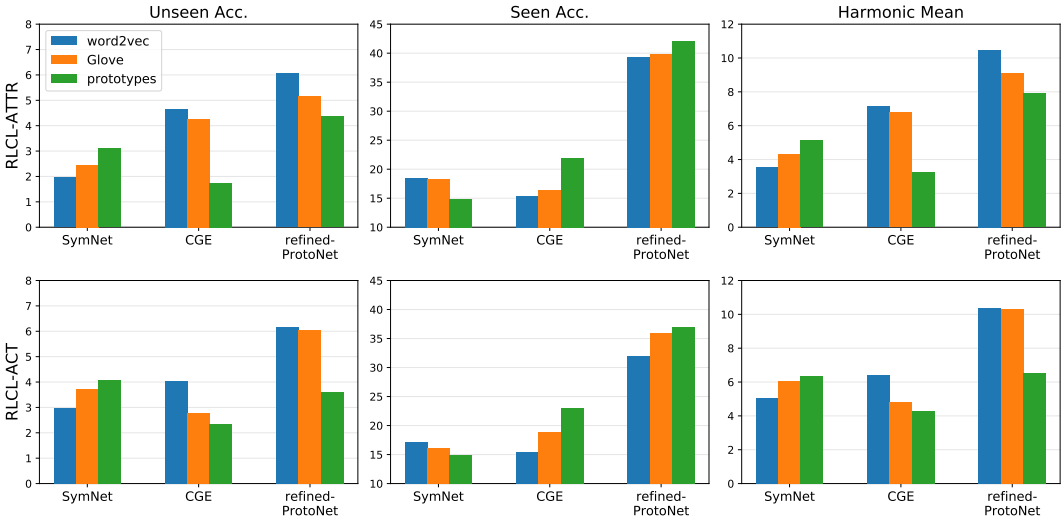

Figure 6: Metrics of SymNet, CGE and our refined-ProtoNet when using different class descriptions on the two datasets.

the various metrics obtained on the validation and test episodes are roughly the same as the training progresses. Therefore, although a better model selection strategy may be explored later for further improvements, we believe that the validation set can currently be used as an alternative to the test environment when selecting the model.

Do the choices of class descriptions affect the performance of CZSL methods? For methods including SymNet, CGE and refined-ProtoNet that require class descriptions as input, we test not only two popular word embedding models word2vec (Mikolov et al., 2013) and Glove (Pennington et al., 2014), but also the prototypes obtained from the support samples to simulate the scenarios where side information is available or unavailable. We see from Table 6 that our refined-ProtoNet achieves the best in all cases, except for the achieved UA using prototypes on RLCL-ACT. An observation worth highlighting is that while the UA and HM performance of CGE and refined-ProtoNet decreased successively with the use of word2vec, Glove, and prototypes, SymNet shows an opposite trend and achieves the best overall performance when using prototypes instead of side information. This suggests that it could serve as a competitive baseline when side information is not available.

Will a deeper backbone improve the performance? Recent few-shot learning studies have reached a consensus that using a deeper convolution backbone may contribute to better performance, and experimental results reported by Chen et al. (2019) demonstrate the reason may be that intra-class variation decreases with the deepening of the backbone. To explore whether a deeper backbone

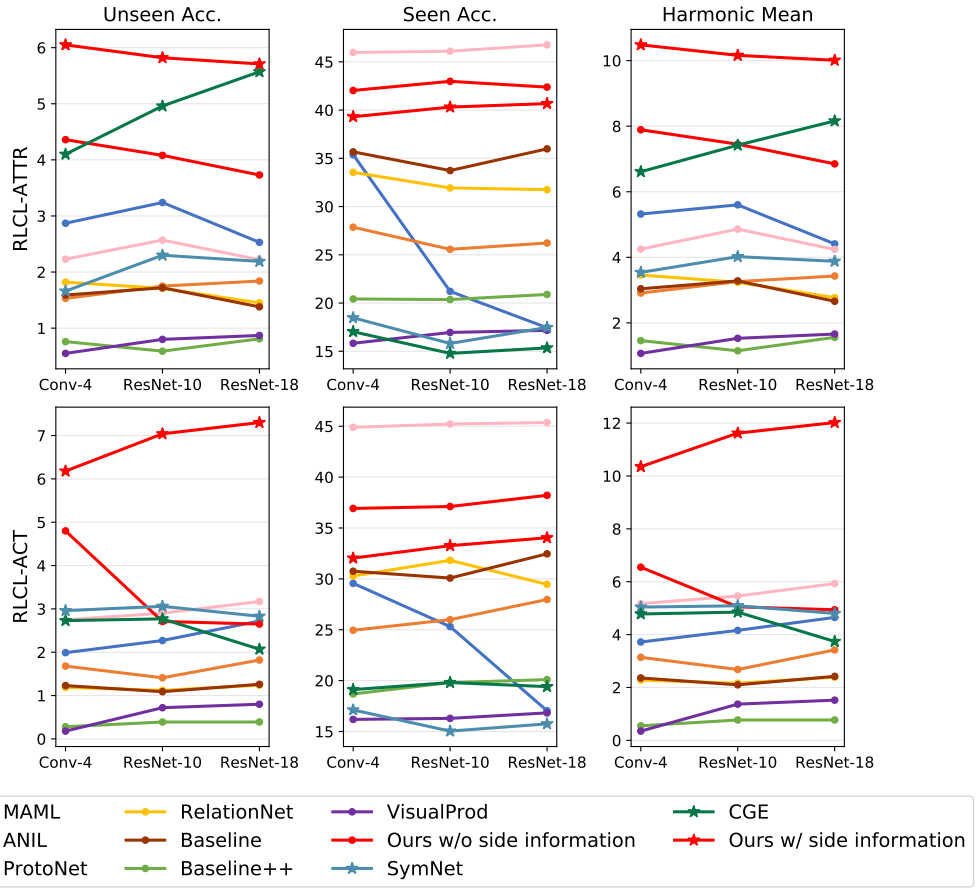

Figure 7: UA, SA and HM results (%) of different backbones on the two datasets. Methods using side information take a pentagram (★) as the marker, and the methods not using side information take a dot (●) as the marker. Our refined-ProtoNet is marked with a red line.

can improve the compositional learners on RLCL, we conduct an experiment that changes the depth of the backbone for all methods when $K_s^c = 5$. The backbones are gradually increased to ResNet-10 and ResNet-18 in comparison with the original Conv-4, which have been the most commonly used in existing works. Specifically, ResNet-18 is the same as described by He et al. (2016) with an input size of 84×84, while ResNet-10 is a simplified version where only one residual building block is used in each layer. We illustrate the results in Figure 7, and unlike most few-shot learning methods, whose performance gets better as the backbone gets deeper, the methods we implemented present a more complicated phenomenon in this experiment. While the tendency of the same method is quite unstable on different datasets, different methods also show no consistent pattern on the same dataset when the backbone deepens. Therefore, a deeper backbone needs to be paired with the appropriate approaches, which can be a valuable research topic in the future.

H Detailed Results for Different Values of K_s^c

In Table 4 and 5, we report the exact values of the results for different values of K_s^c for reference.

Table 4: Detailed results (%) for different values of K_s^c on RLCL-ATTR.

Method	RLCL-ATTR				
	Unseen Acc.	Seen Acc.	Harmonic Mean	Primitive 1 HM	Primitive 2 HM
			$K_s^c = 5$		
MAML	2.87±0.50	35.38±1.69	5.32±0.87	29.50±0.71	40.09±3.06
ANIL	1.53±0.17	27.86±0.52	2.91±0.31	25.93±2.15	33.38±3.02
ProtoNet	2.23±0.99	45.97±1.83	4.25±1.80	32.02±1.90	48.01±1.79
RelationNet	1.82±0.39	33.55±1.26	3.46±0.69	28.26±2.51	38.36±1.92
Baseline	1.59±0.35	35.67±0.63	3.04±0.65	28.95±1.47	38.47±2.05
Baseline++	0.76±0.10	20.42±0.33	1.46±0.19	23.84±1.27	25.46±4.14
VisualProd	0.55±0.45	15.83±0.60	1.07±0.83	22.80±2.59	22.90±1.31
refined-ProtoNet	4.36±0.84	42.03±1.58	7.89±1.36	31.58±1.81	48.48±1.19
Learning with Side Information					
SymNet	1.96±0.95	18.47±0.68	3.54±1.54	27.24±2.03	24.47±2.57
CGE	4.10±1.09	17.03±0.13	6.61±1.43	25.06±0.20	31.04±0.84
refined-ProtoNet	6.05±1.64	39.31±0.73	10.48±2.44	33.14±2.27	49.19±1.75
			$K_s^c = 3$		
MAML	3.04±0.59	31.00±0.70	5.53±0.97	28.69±2.19	37.17±2.20
ANIL	1.44±0.39	25.41±1.49	2.73±0.70	26.77±1.95	30.46±1.43
ProtoNet	2.11±0.68	41.69±0.90	4.01±1.22	30.58±0.57	44.21±2.10
RelationNet	1.75±0.19	29.51±3.15	3.30±0.36	27.98±1.68	34.43±3.55
Baseline	1.62±0.62	32.59±0.77	3.09±1.13	28.16±2.19	36.79±3.01
Baseline++	0.33±0.13	18.59±0.15	0.65±0.25	24.75±1.89	22.41±4.34
VisualProd	0.87±0.10	17.27±0.78	1.65±0.18	23.02±2.81	24.29±2.45
refined-ProtoNet	3.78±1.16	37.20±0.83	6.85±1.91	30.05±1.39	44.61±2.45
Learning with Side Information					
SymNet	2.02±0.15	17.68±0.43	3.62±0.25	26.93±0.53	24.19±1.51
CGE	5.34±0.24	13.99±0.57	7.73±0.26	25.98±1.11	31.02±1.21
refined-ProtoNet	6.68±1.02	34.93±3.58	11.21±1.40	32.32±4.29	46.80±2.17
			$K_s^c = 1$		
MAML	2.05±0.72	19.72±0.58	3.70±1.17	26.04±0.06	27.02±2.42
ANIL	1.77±0.22	19.00±1.57	3.23±0.36	24.40±1.30	26.66±2.14
ProtoNet	1.70±0.47	30.07±1.58	3.22±0.85	27.74±1.11	36.15±1.59
RelationNet	1.59±0.22	22.22±1.18	2.96±0.38	25.81±1.28	29.41±1.05
Baseline	1.21±0.32	23.44±0.25	2.31±0.58	26.07±0.64	29.00±1.18
Baseline++	0.69±0.08	18.64±0.40	1.33±0.15	24.66±1.33	23.46±1.99
VisualProd	1.24±0.43	20.99±0.18	2.34±0.76	25.44±2.15	27.41±1.79
refined-ProtoNet	2.73±0.36	27.37±1.77	4.97±0.56	28.73±0.66	35.30±1.57
Learning with Side Information					
SymNet	1.94±0.08	17.34±0.80	3.48±0.12	27.01±1.05	23.95±2.87
CGE	4.65±1.12	15.40±0.54	7.13±1.29	25.97±3.30	31.56±1.26
refined-ProtoNet	5.87±1.74	26.79±2.29	9.62±2.45	29.29±1.98	42.44±4.22

Table 5: Detailed results (%) for different values of K_s^c on RLCL-ACT.

Method	RLCL-ACT				
	Unseen Acc.	Seen Acc.	Harmonic Mean	Primitive 1 HM	Primitive 2 HM
			$K_s^c = 5$		
MAML	1.99±0.27	29.56±2.03	3.72±0.45	25.94±1.02	29.26±1.71
ANIL	1.68±0.51	24.95±2.95	3.14±0.88	25.70±4.61	25.03±3.50
ProtoNet	2.75±0.90	44.89±1.31	5.17±1.60	45.21±2.52	35.03±1.42
RelationNet	1.19±0.46	30.28±5.55	2.28±0.84	31.36±6.31	27.91±2.06
Baseline	1.23±0.20	30.75±1.23	2.36±0.37	28.21±1.67	27.69±2.26
Baseline++	0.28±0.04	18.69±1.17	0.55±0.08	13.50±2.30	22.42±2.36
VisualProd	0.18±0.14	16.19±0.32	0.35±0.28	13.72±1.80	21.88±2.73
refined-ProtoNet	3.60±0.89	36.92±1.57	6.55±1.45	39.20±2.96	33.37±2.24
Learning with Side Information					
SymNet	2.96±0.35	17.12±0.53	5.04±0.53	31.74±1.64	22.30±2.03
CGE	2.73±0.78	19.12±0.65	4.78±1.17	25.57±2.48	23.05±1.11
refined-ProtoNet	6.18±1.20	32.04±4.78	10.35±1.65	41.84±3.12	33.00±1.87
			$K_s^c = 3$		
MAML	2.44±0.69	23.55±1.46	4.41±1.13	23.98±1.32	30.28±1.93
ANIL	1.71±0.39	22.91±1.23	3.19±0.68	24.41±1.03	25.55±0.32
ProtoNet	2.55±0.62	40.51±1.31	4.79±1.10	41.20±0.72	33.24±1.41
RelationNet	1.24±0.13	27.81±0.71	2.38±0.23	28.39±2.22	27.02±3.69
Baseline	1.26±0.47	27.22±1.29	2.40±0.87	24.57±3.76	26.88±3.19
Baseline++	0.17±0.21	17.33±0.10	0.33±0.42	13.43±0.62	21.61±3.71
VisualProd	0.65±0.87	17.46±0.68	1.25±1.64	20.30±3.28	22.47±2.22
refined-ProtoNet	2.77±0.41	33.62±1.57	5.12±0.68	34.64±1.05	31.30±1.35
Learning with Side Information					
SymNet	2.88±0.05	17.13±0.41	4.94±0.05	31.96±1.09	21.74±1.88
CGE	3.88±0.48	16.89±0.73	6.31±0.59	27.39±0.80	24.88±1.41
refined-ProtoNet	6.99±1.42	29.09±3.38	11.26±1.66	40.25±0.94	33.34±2.02
			$K_s^c = 1$		
MAML	1.24±0.31	18.36±0.41	2.33±0.55	16.60±0.85	27.05±3.45
ANIL	1.45±0.42	19.44±0.83	2.70±0.73	22.39±2.36	25.63±1.98
ProtoNet	1.78±0.30	29.96±1.03	3.36±0.53	33.37±1.80	28.70±1.25
RelationNet	0.87±0.52	20.06±3.30	1.66±0.96	19.49±5.32	24.36±3.25
Baseline	0.71±0.20	20.82±0.53	1.37±0.38	19.70±2.54	24.07±1.82
Baseline++	0.32±0.48	18.02±0.25	0.63±0.93	14.32±3.85	22.69±1.32
VisualProd	0.88±0.19	21.97±0.56	1.68±0.36	26.12±1.78	25.43±1.15
refined-ProtoNet	1.80±0.33	23.39±0.36	3.35±0.56	25.17±0.19	28.56±0.89
Learning with Side Information					
SymNet	2.28±1.71	17.90±0.56	4.01±2.72	27.35±1.59	23.02±2.82
CGE	4.05±0.78	15.51±0.91	6.41±0.91	28.56±2.09	26.39±1.50
refined-ProtoNet	6.41±2.02	22.57±1.67	9.96±2.46	34.69±4.19	31.30±2.66