Known Unknowns: Learning Novel Concepts Using Reasoning-by-Elimination

Harsh Agrawal[1,2] **Eli A. Meirom**[1] **Yuval Atzmon**[1] **Shie Mannor**[1] **Gal Chechik**[1]

[1]NVIDIA Research, Israel
[2]Georgia Tech, Georgia, USA

Abstract

People can learn new visual concepts without any samples, from information given by language or by deductive reasoning. For instance, people can use *elimination* to infer the meaning of novel labels from their context. While recognizing novel concepts was intensively studied in zero-shot learning with semantic descriptions, training models to learn by elimination is much less studied. Here we describe the first approach to train an agent to reason-by-elimination, by providing instructions that contain both familiar concepts and unfamiliar ones ("*pick the red box and the green wambim*"). In our framework, the agent combines a perception module with a reasoning module that includes internal memory. It uses reinforcement learning to construct a reasoning policy that, by considering all available items in a room, can make a correct inference even for never-seen objects or concepts. Furthermore, it can then perform one-shot learning and use newly learned concepts for inferring additional novel concepts. We evaluate this approach in a new set of environments, showing that agents successfully learn to reason by elimination, and can also learn novel concepts and use them for further reasoning. This approach paves the way to handle open-world environments by extending the abundant supervised learning approaches with reasoning frameworks that can handle novel concepts.

1 INTRODUCTION

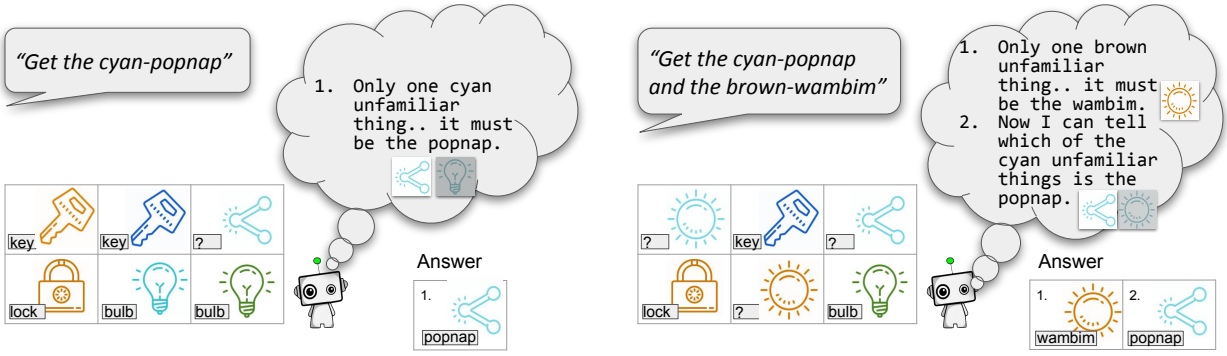
Machine learning models are primarily trained to perform inductive reasoning: generalizing rules from training examples. In comparison, people routinely use other forms of reasoning, including various forms of deduction. For example, young infants can use reasoning-by-elimination (disjunctive syllogism) to interact with an unfamiliar object and for learning new concepts [Cesana-Arlotti et al., 2020, Grigoroglou et al., 2019, Markman, 1990, Markman et al., 2003]. This type of reasoning with unfamiliar objects is particularly important for open-world scenarios where agents experience a mix of familiar and unfamiliar objects. Unfortunately, existing SOTA methods for object recognition struggle with novel and out-of-distribution items.

Despite its apparent utility, it is not known if and how deductive reasoning mechanisms like elimination can be learned from data with current algorithms or if it can be employed within the framework of statistical machine learning. The current paper explores this problem and describes a framework for training an agent to reason-by-elimination following short language instructions in a simple synthetic world.

Reasoning by elimination (RBE), formally known as *disjunctive syllogism* or modus tollendo ponens, is a fundamental mode of reasoning in classical logics [Hurley, 2014]. It has been lauded in popular literature as an admirable form of deduction [Conan Doyle, 1932] and investigated in developmental psychology as a reasoning mechanism applied by infants [Cesana-Arlotti et al., 2020, Grigoroglou et al., 2019, Mody and Carey, 2016, Markman, 1990, Markman et al., 2003] and animals [Fugazza et al., 2021].

In the simplest form of RBE, an agent is faced with two options, P or Q, one of which must be correct. It is then told that P is incorrect and has to infer that Q is correct. It has been argued that this setting occurs naturally during language learning. When a parent addresses a child with a phrase that contains an unknown word, like "take the X", the child can deduce that X stands for a novel object, rather than a familiar one [Halberda, 2006] (but see [Xu and Tenenbaum, 2007] for a Bayesian alternative). In a more advanced setup, once the new concept has been identified, the agent can learn with this new example (one-shot learning), and use it to further identify new concepts by applying elimination again. Elimination can therefore provide a major path to reason in an open-world setting.

Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI 2021), PMLR 161:504–514.



(a) Single Object Task (b) Double Object Task

Figure 1: Illustration of **reasoning-by-elimination**. **(a) Single Object Task**: An agent receives an instruction to pick up an unfamiliar object. The agent has never seen a "popnap" before but has seen cyan-colored objects before. It has also seen a bulb before. The agent needs to realize that there is only one cyan-colored object with an unseen shape so it must be the object being referred to in the instruction. **(b) Two-Object Task**: An agent receives an instruction to pick up two objects in a room, where the objects are described by unfamiliar concepts. The agent must deduce by elimination which objects to choose. In the example, the agent is asked to retrieve a cyan-*popnap* and a brown-*wambim*, where both *popnap* and *wambim* refer to unknown shapes: First, it finds that there is only one unfamiliar brown object (bottom-row middle) and two unfamiliar cyan objects, and therefore it chooses the unfamiliar brown object for the brown-wambim. At this point it makes a 1-shot learning action, and updates its perception module about the concept of *wambim*. Once the agent can recognize the wambim, it can eliminate the cyan-wambim and choose the top-right object to be the cyan-popnap. See section 3 for details.

In this paper, we show how agents can be trained using reinforcement learning to make disjunctive syllogism inferences, use them to learn new concepts, and apply their novel knowledge to reason about other new concepts. We designed a learning setup where an agent wanders in a synthetic scene and has to follow simple instructions (see Figure 1, "Get the cyan popnap"). The scene contains objects, some of which have been encountered in a pre-training phase, and some are novel. In the simplest setup (Figure 1a), the agent action space consists of picking any object in the scene and the agent learns to use elimination to pick an object based on the textual instruction. In a more advanced setup (Figure 1b), the agent action space also includes active one-shot learning of the new concept, and the agent learns to use it to infer the identity of a second object. The agent follows three steps: *eliminate*, *learn*, *eliminate*. Each "eliminate" step infers a novel class without being trained with labeled samples. The "learn" step applies one-shot learning, namely, learn a new concept from a single observation.

This novel learning setup can be viewed as a new kind of zero-shot learning. Novel objects are recognized through inference about a group of objects. Unlike classical zero-shot learning [Lampert et al., 2009, Romera-Paredes and Torr, 2015, Atzmon and Chechik, 2019], no prior information is available about novel objects. Here we built a compositional space of concepts, defined by color and shape, which makes it easier to generate novel combinations from a small vocabulary. Such zero-shot compositional generalization is encountered in larger and more natural datasets [Atzmon et al., 2020, Johnson et al., 2017].

Our main novel contributions are: (1) A formulation of a novel learning problem where an agent has to learn to reason by elimination. (2) A system for training an agent using reinforcement learning to reason by elimination. (3) We show how the agent can be trained to actively perform one-shot learning for learning new concepts, and use the new concepts for further reasoning by elimination (4) Evaluations on a dataset of compositional instructions, comparing various components of the model.

2 RELATED METHODS

Word learning has been studied in cognitive psychology, with a focus on how infants learn the meaning of new words. It has been shown that children can use elimination for that goal, sometimes applying a principle of "mutual exclusivity" [Cesana-Arlotti et al., 2020, Grigoroglou et al., 2019, Mody and Carey, 2016, Markman, 1990]. Situated word learning has also been explored in the contxt of deep models. Hill et al. [2017] studied an agent exploring a 3D room and rewarded when its actions match instructions. There is also large literature on **grounding language** in vision and action [Reckman et al., 2010, Tellex et al., 2011, Chen and Mooney, 2011, Duvallet et al., 2013, Hemachandra et al., 2015, Andreas and Klein, 2015, Misra et al., 2017a, Jänner et al., 2018, Anderson et al., 2018, Krantz et al., 2020, Qi et al., 2020, Blukis et al., 2019, 2020]. See Luketina et al. [2019] for a detailed survey. Among these, the most relevant work is by Blukis et al. [2020], who studied a few-shot language-conditioned object grounding task. There, as the

agent navigates the environment, it aligns natural-language mentions within instructions to novel objects in the environment using an extensible database. The agent infers novel objects using navigation directions and by spatial reasoning with respect to other object mentions in the instructions. In our setup, the agent has to rely on reasoning-by-elimination.

Most relevant to this paper is the recent work by Hill et al. [2021]. It shows that after a single introduction to a novel object via visual perception and language ("This is a dax"), the agent can manipulate the object as instructed ("Put the dax on the bed") in a simulated 3D world. Its architecture combines within-episode 1-shot ("fast-mapping') learning, with long-term knowledge, when trained with conventional RL algorithms. Our paper takes a step forward: (1) Rather than having an explicit supervision that is paired with the novel concept, our supervision is implicit, reasoning about a novel concept from a group of objects. (2) Our policy can reason about multiple novel concepts and resolve ambiguities. (3) Our visual-textual concepts are compositional.

Zero-shot Learning (ZSL): The standard zero-shot learning setup [Xian et al., 2017, Lampert et al., 2009, Atzmon and Chechik, 2019, 2018], a classifier recognizes novel classes based on an auxiliary semantic class description ("A zebra has black and white stripes"). The problem setup in our paper differs from standard ZSL in two ways: (1) No description is provided about the novel classes. Novel classes are nevertheless inferred without any labeled samples. (2) Novel objects are recognized through RBE inference about a group of objects.

Compositional ZSL: The compositional ZSL setup trains a classifier to recognize new combinations of known components [Misra et al., 2017b, Nagarajan and Grauman, 2018, Purushwalkam et al., 2019, Atzmon et al., 2020, Mancini et al., 2021, Naeem et al., 2021], like colors and shapes. Namely, the training data includes examples of *all* visual concepts in a vocabulary, but only a subset of their pairs. For instance, the training data may consist of images of red apples and green peppers, while the test data includes images of green apples. The compositional setup here is different, because some elements in the vocabulary are not at all included in the training data.

Out-of-distribution detection: Reasoning about a single unfamiliar object resonates with out-of-distribution detection (OOD), where images of novel concepts may be considered as "out-of-distribution" samples. There is a large body of work on 1-class and anomaly detection which we do not survey here. In the current context, the most relevant works include [Hendrycks and Gimpel, 2017, Atzmon and Chechik, 2019, Vyas et al., 2018], by detecting an OOD sample if the largest softmax score is below a threshold and training the OOD detection models on a set of "leave-out" classes. Part of the architecture of the policy network is inspired by these ideas. The setup in our paper is different from OOD detection in that the policy we learn can (1) resolve ambiguities when reasoning about multiple novel concepts at a time; (2) novel concepts can be visually entangled with familiar concepts (e.g. in brown-wambim, brown is familiar, while wambim is novel); and (3) The policy learns when to apply a one-shot learning step in order to extend its vocabulary about the novel concept it discovered.

Predicate logic: Learning a policy for reasoning by elimination may be modelled with soft predicate logic applied as constraints or potentials [Richardson and Domingos, 2006, Kimmig et al., 2012, Bach et al., 2017] or with an end-to-end differentiable deep network [Atzmon and Chechik, 2018, Wang and Poon, 2018]. However, such an approach requires an additional source of external knowledge in the form of logic predicates. Our approach is different as it does not require external knowledge when learning the policy.

3 AN ENVIRONMENT FOR REASONING BY ELIMINATION

We designed a new environment for evaluating reasoning by elimination. The environment combines visual, textual, and action modalities; see Figure 1. In each episode, an agent enters a room that contains a set of objects $\mathcal{O}$, and is given an instruction T for choosing one or two of the objects in the room. Each object $o \in \mathcal{O}$ is described by two attribute properties, a "shape" (o^s) and a "color"(o^c).

Some of the shape and color concepts are known in advance ("seen") to the agent while other concepts are unknown ("unseen"). An object may exhibit any combination of seen or unseen concepts. For example, a *cyan-key*, where both concepts, cyan and key, are seen ("familiar"); a *cyan-popnap*, with a previously unseen shape ("popnap"); or a *smaragdine-popnap*, where both shape and color were not seen before.

The instruction $T = (T_1, T_2 ... T_m)$ contains the shape and color attribute for each one of the goal objects $T_i = \{T_i^S, T_i^C\}, i = 1..m$. The goal of the agent is to select the objects specified in the instruction. We have studied the cases with $m = 1$ and $m = 2$, but the extension to more complicated instructions follows naturally.

Action space: At each time step, the agent can either call a `"NEXT"` action and move to the next object, or call a `"CHOOSE #k"` action to indicate it found the object in the $k-$th part of the instruction. Once all objects are chosen, the episode ends. When all $|\mathcal{O}|$ objects in the episode were observed, the `"NEXT"` action cycles back to the first object.

With this setup, we designed two reasoning and learning challenges for the agent.

3.1 SINGLE-OBJECT TASK

Reward: In the first task, the goal of the agent is to select a single object by matching the given textual description. When the chosen object matches the instruction, the agent receives a positive reward and receives no reward if there is a mismatch. To encourage the agent to solve the task as quickly as possible, it receives a (negative) slack penalty after each action.

Episode creation: Environments are generated with five objects. Each object is assigned a shape and a color. Some objects in the environment are familiar ("seen") in the sense that the agent was pretrained to match their visual representation to a textual description, others are novel ("unseen"). In each room, there is only a single object that can match the textual description. It can either be a seen object or an unseen object. See examples in Fig. 1a. When the instruction refers to a familiar object, the agent has to learn to pick the object that matches the description. When the instruction refers to an unseen object (either with a novel color, or shape, or both), the agent has to deduce by elimination which unseen object relates to the novel description.

To make the task solvable, the set of objects obeys some constraints. For example, given an instruction "cyan-popnap", whose shape description ("popnap") was not encountered before, the set of objects can not include both "cyan-popnap" and "cyan-wambim" as it is not possible to distinguish between two unseen shapes ("popnap" and "wambim"). In general, if the text instruction contains an unseen attribute (e.g., unseen shape), the environment will contain just one object with the corresponding unseen attribute and the matching complementary attribute (e.g., color) in the object set. Likewise, if both the shape and the color in the instruction were unseen before, the episode can not contain two objects where both attributes were unseen before.

3.2 TWO-OBJECT TASK

The second task extends the single-object task in a fundamental way, by adding a **one-shot learning** component. First, the agent has to infer one object using elimination. Then, it has to use its decision as a supervision signal for learning the new concepts. Finally it needs to select a second novel object which can only be detected correctly based on the first novel object. The agent therefore has to perform a combination of **Eliminate-Learn-Eliminate** steps which combines reasoning with one-shot-learning.

Reward: Same as in single-object task, but here the agent is rewarded when recognizing both novel concepts.

Episode creation: In each episode, instructions require the agent to retrieve two objects with novel (previously unseen) attributes from a set with a total of five objects (Figure 1). Each environment may contain up to three visual objects with one novel property. Two of the novel objects share a seen property (Figure 1: cyan-popnap and cyan-wambim), and the third novel object can be used to resolve the ambiguity between the first two (Figure 1: brown-wambim). The agent has to deduce by elimination which is the non-ambiguous object, and then use that knowledge in order to disambiguate between the objects that share the seen property. For instance, given a set of five objects in Figure 1, and the instruction "Get the cyan-popnap and the brown-wambim", the agent can easily learn what a "wambim" is because there is only one unseen shape that has a "brown" color. Having learned what a "wambim" is, the agent can now disambiguate between "cyan-popnap" and "cyan-wambim" to retrieve the "cyan-popnap". Naturally, we also allow the instruction to specify the easier case of retrieving familiar objects.

Notation: Below, we denote the set of all shape labels by $\mathcal{S}$, all colors by $\mathcal{C}$, and set of all object instances present in the environment by $\mathcal{O}$. We use subscript notation to denote the subset of *seen* e.g. $\mathcal{C}_{seen}$ or *unseen* (novel) concepts.

4 APPROACH

The goal of the agent is to reason about an instruction and choose the correct (seen or unseen) objects in a room containing both seen and unseen objects. Consider first the most naïve approach, where one learns a policy based solely on the visual representation of the objects observed in the episode and the textual instruction representation. This approach succeeds if the target object was seen during training, but fails if this is not the case. Such representation can not reason about new, unseen objects.

Our proposed architecture allows an agent to reason-by-elimination in a multi-modal environment, and progressively learn novel concepts. These new deduced concepts can be used in an online manner when reasoning about multiple novel concepts.

The agent has two key components: a perception module – which casts the textual instruction and the visual objects presented in the episode to a shared embedded representation, and a reasoning policy, which retrieves the correct objects. The reasoning module uses this embedding to infer which unseen object in the environment must match with the novel textual description using elimination.

Next, we describe each component in more detail. First, we consider the *Single-Object* Task. Later, we extend our approach by augmenting the action space with one-shot learning for the *Two-Object* Task.

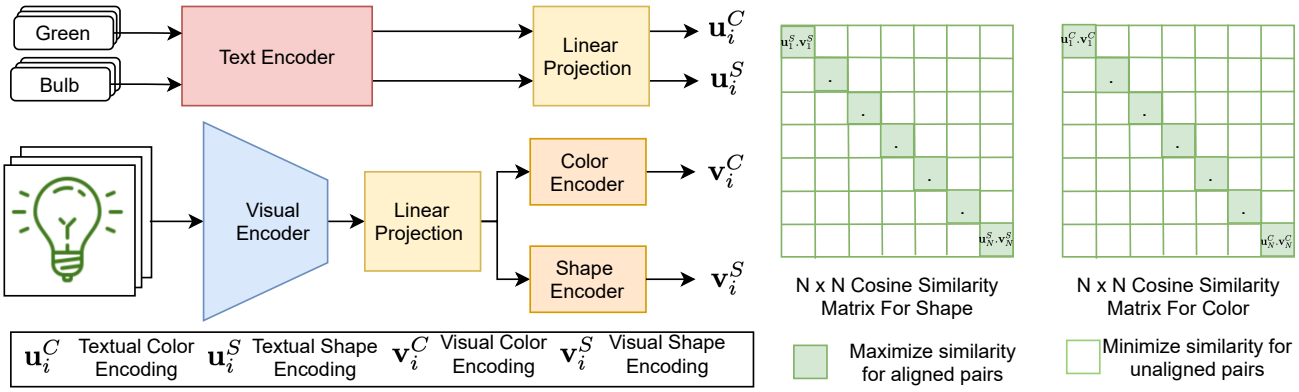


Figure 2: The Perception Module. It casts both the textual instruction and the visual appearance of an object into a joint multi-modal embedding space. The two mappings are learned by maximizing the cosine-similarity between the visual and textual feature vectors using symmetric contrastive loss.

4.1 THE PERCEPTION MODULE

The perception module is designed to cast both the textual instruction and the visual appearance of an object into a joint multi-modal embedding space (Figure 2). It is composed of a visual encoder, which take as input the object image and cast it to shape and color representations, and text encoder, which maps each attribute to a feature vector of the same dimension. The mappings are learned by maximizing the cosine-similarity between the visual and textual feature vectors using symmetric contrastive loss [Zhang et al., 2020]. See supplementary for additional details.

The visual encoder receives as input an object image and processes it with a 3-layer convolutional neural network backbone to obtain a visual representation of the object $\mathbf{x} \in \mathbb{R}^{512}$. $\mathbf{x}$ is then projected into two separate representations: $\mathbf{v}^S \in \mathbb{R}^{32}$ representing the shape, and $\mathbf{v}^C \in \mathbb{R}^{32}$ representing the color.

The text encoder processes the textual instruction given to the agent. We encode each attribute label (color or shape) with a pretrained BERT model [Devlin et al., 2019] followed by a non-linear projection to $\mathbb{R}^{32}$. This yields a dense feature vector for the color-label $\mathbf{u}^C$ and for the shape-label $\mathbf{u}^S$.

Losses and training details. We pre-train the perception module to recognize the vocabulary of seen concepts. During the pre-training stage, we sample a mini-batch of N aligned image-text pairs such that all the concepts occurring in the batch are unique. We learn to maximize the cosine similarity between the visual and textual representations of the same object, using a sum of symmetric contrastive loss functions [Zhang et al., 2020] for both the shape and color. For an image indexed by i in a batch of N aligned image-text feature pairs $\{\mathbf{v}_k, \mathbf{u}_k\}_{k=1}^N$, the symmetric contrastive loss is defined by

$$\ell\big(i; \{\mathbf{v}_k, \mathbf{u}_k\}_{k=1}^N\big) = -\log \frac{\exp\big(\langle \mathbf{v}_i, \mathbf{u}_i \rangle\big)}{\sum_{k=1}^N \langle \mathbf{v}_i, \mathbf{u}_k \rangle} - \log \frac{\exp\big(\langle \mathbf{u}_i, \mathbf{v}_i \rangle\big)}{\sum_{k=1}^N \langle \mathbf{u}_i, \mathbf{v}_k \rangle}, \tag{1}$$

where $\langle \mathbf{u}, \mathbf{v} \rangle$ denotes the cosine similarity.

The perception loss for a minibatch of N aligned image-text pairs, takes the color image-text representations $\{\mathbf{u}^C{}_k, \mathbf{v}^C{}_k\}_{k=1}^N$, shape representations $\{\mathbf{u}^S{}_k, \mathbf{v}^S{}_k\}_{k=1}^N$, and averages the contrastive losses across the mini-batch samples:

$$\mathcal{L}_{PRCP} = \frac{1}{N} \sum_{i=1}^{N} \ell\big(i; \{\mathbf{u}^C{}_k, \mathbf{v}^C{}_k\}_{k=1}^N\big) + \ell\big(i; \{\mathbf{u}^S{}_k, \mathbf{v}^S{}_k\}_{k=1}^N\big). \tag{2}$$

4.2 POLICY NETWORK

The policy network learns to reason about the instruction in the environment, to retrieve the matching objects.

For all *known* concepts, a policy can retrieve the object described in an instruction, just by choosing the one that maximizes the cosine-similarity in the vision-language embedding space, while the irrelevant (known) objects will yield low-magnitude similarities with the instruction. However, when an instruction refers to an unseen concept, its textual embedding will not match *any* of the objects in the room, neither known nor unknown.

In particular, a textual description containing an unknown attribute will yield a low-magnitude similarity score with the visual representation (the output of the visual encoder) of every object in the environment, seen or unseen, whether it is the goal object or not. Hence, without proper reasoning, all objects are likely to be chosen uniformly.

Policy Architecture. We propose an architecture (Figure 3) that can provide additional information to the policy network, allowing it to deduce that an *observed object* consists of unknown properties and that the given *instruction* refers to unknown concepts.

Inputs. The policy gets as input the output of the perception module: a feature representation of the current object image

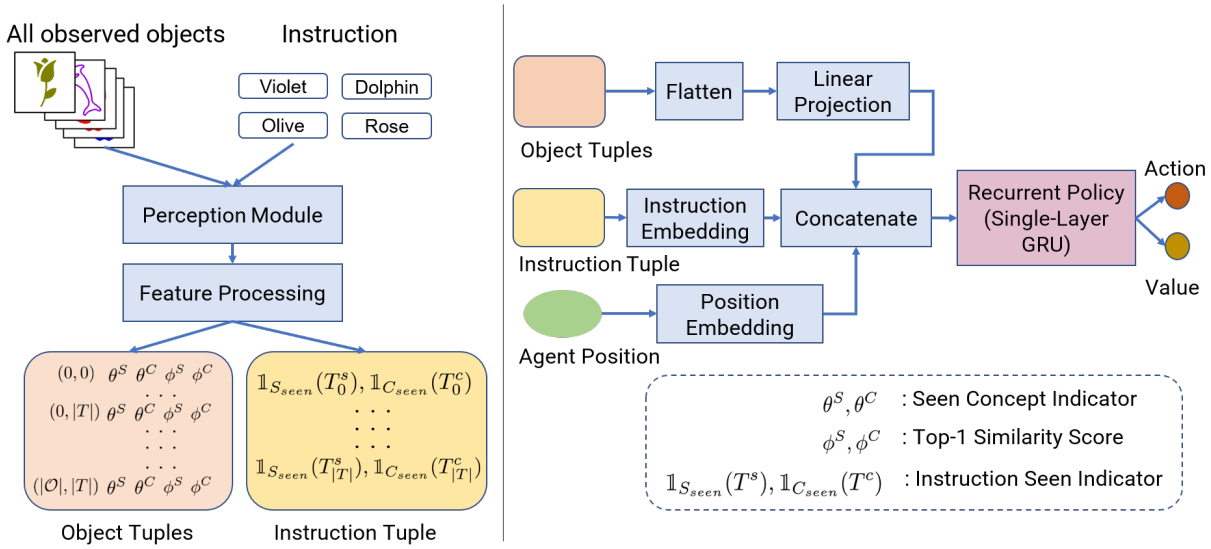


Figure 3: **Reasoning Module**. Left - The recurrent module takes as input a 4-dimensional tuple describing the objects, an instruction tuple indicating which concepts in the instructions are seen, and the agent's position in the environment. Right - The input is fed to a single-layer GRU which outputs an action probability distribution and an estimate of the value function.

and the textual instruction. Additionally, the policy stores a dictionary of all the textural representations for each attribute (shape or color) seen during the pretraining stage. The information is used to form an internal state representation, which incorporates seen concept indicators.

Seen concept indicator. As it is difficult to distinguish an unseen object just by its similarity score with the textual instruction, we construct an additional "seen indicator". This indicator is the top score of the object visual embedding with all the corresponding seen textual attributes (color / shape) encoding. Ideally, this should be close to one for images of seen attributes, while providing a significantly lower score for images of unseen concepts. For example, the visual shape encoding v^S of "blue key' should be similar to the textual encoding u^S of a "key" if keys were part of the seen training set. Similarly, the visual shape encoding of a "brown wambim' v^S would not be similar to any textual encoding, when a "wambim" is excluded from the seen training set vocabulary.

Internal state. The policy holds an internal state representation, which is a $4 \times |\mathcal{O}| \times m$-dimensional tuple. It is initialized as a zeros tuple, and for each encountered object o and each goal object T_i the agent constructs a 4-tuple as follows:

1. The first term is the similarity term $\phi^S(o,i) = \langle \mathbf{v}^S{}_o, \mathbf{u}^S{}_i \rangle$ between the representation of the object shape $\mathbf{v}^S{}_o$ and the textual representation of each shape mentioned in the instruction $\mathbf{u}^S{}_i$.
2. Similarly, the second term is the similarity $\phi^C(o,i) = \langle \mathbf{v}^C{}_o, \mathbf{u}^C{}_i \rangle$ between the representation of the object color $\mathbf{v}^C{}_o$ and the color representation of each object mentioned in the instruction $\mathbf{u}^C{}_i$.
3. The top similarity score of shape representation of the visual object with all the shape embeddings in the vocabulary. This is the value of the seen shape indicator. Concretely, we compute:

$$\theta^S(o) = \max\{\langle \mathbf{v}^S{}_o, \mathbf{u}^S{}_j \rangle : j \in \mathcal{S}\} \tag{3}$$

4. Similarly, we compute the top similarity score for the color (seen color indicator) as follows:

$$\theta^C(o) = \max\{\langle \mathbf{v}^C{}_o, \mathbf{u}^C{}_j \rangle : j \in \mathcal{C}_{seen}\}. \tag{4}$$

To summarize, for every goal object in the instruction $T_i \in T$, each observed object $o \in \mathcal{O}$ is represented by a 4-dimensional tuple $\{\phi^S(o,i), \phi^C(o,i), \theta^S(o), \theta^C(o)\}$. Thus, for the single-object task ($m = 1$), each observed object is represented by a 4-dimensional tuple and 8-dimensional tuple for two-object task ($m = 2$).

In addition, the policy is aware of the agent position in the environment $p \in [1, |O|]$, and a $2m$-dimensional instruction embedding indicating whether the shape and color attribute mentioned in the textual description are novel. $\{\mathbb{1}_{\mathcal{S}_{seen}}(T_i^s), \mathbb{1}_{\mathcal{C}_{seen}}(T_i^c)\}, \forall T_i \in T$

Output. The policy module constructs the internal state tuple, and outputs a score for each action. During training, these scores are converted to a probability distribution using a softmax function, while during inference the highest scored action is chosen.

4.3 THE TWO-OBJECT TASK: 1-SHOT LEARNING OF NEW CONCEPTS

While the single-object task requires to reason about a *single* out-of-distribution object, the two-object task is harder. It requires the agent to disambiguate the first object and then use that newly acquired knowledge to disambiguate the second object.

One approach would be to memorize the first chosen object using an external memory [Hill et al., 2021], and build an architecture that can reason about it with respect to the remaining objects in the room, before choosing the second object. Such architecture may be complex to construct, and would be hand-crafted for this specific task.

Instead, in our approach we take a simpler alternative: We *augment* the action space of the agent with two additional actions, "`CHOOSE & LEARN CONCEPT FROM #k`" for $k = 1, 2$. This action performs a one-shot learning action that updates the weights of the perception module with respect to the $\texttt{k}^{th}$ instruction. Sequentially, it calls "`CHOOSE #k`" at the environment. (See the '`step`' function in Listing 1)

The one-shot learning action updates the perception module to match the novel text term with the novel visual attribute of the object. This action affects the internal representation in the agent model and does not affect the physical environment. For simplicity, we implement it by a simple sequence of gradient updates with respect to the perceptual loss in Eq. (2) using the few encountered seen objects in the episode so-far. The seen concepts in the episode are the object with high seen concept indicators (> 0.9) both for the shape and color concepts. The update step is described in the function '`update`' in Listing 1. Note that these updates only affect the perception module of the agent.

With these update actions, the agent can now learn a novel concept from the first chosen object, and use that knowledge to disambiguate the second object. Note that if the agent applies the update action incorrectly, it is penalized by a wasted turn. Furthermore, such incorrect updates are noisy (there are just a few, usually two negative samples for the contrastive loss) and those updates may deteriorate the perception module accuracy considerably.

As each "`CHOOSE & LEARN CONCEPT FROM #k`" action requires gradient updates, which are computationally heavy, in practice, we limit the number of such actions to 2 per episode.

Listing 1: Pseudo-code describing the agent policy.

```
def update(perception_module, env, k):
  # Update the perception module for
  ↪  learning a new concept.
  current_obj = env.get_current_object()
  ins_obs = instruction[k] # kth object
  ↪  features.
  if ins_obs.shape == "unseen":
    update_attr = "shape"
  else:
    update_attr = "color"

  # Find seen objects and associated labels
  ↪  by checking the similarity score of
  ↪  each observed object against the seen
  ↪  concepts vocabulary.
  seen_objects, seen_object_labels =
  ↪  perception_module(env.get_observation(),
  ↪  update_attr)
  batch_vis = seen_objects + [current_obj]
  batch_labels = seen_object_labels +
  ↪  [ins_obs[update_attr]]
  visual_emb, text_emb =
  ↪  perception_module(batch_vis,
  ↪  batch_labels)
  loss = contrastive_loss(visual_emb,
  ↪  text_emb)
  loss.backward()

def take_action(env, action):
  if action in ["next", "choose #1",
  ↪  "choose #2"]:
    reward = env.step(action)
  elif action matches "choose and learn
  ↪  #i":
    reward = env.step("choose #i")
    update(perception_module, env, i)
    # Update input features of the policy
    ↪  based on the new perception module.
    observations =
    ↪  perception_module.observe(env)
 return reward
```

5 EXPERIMENTS

Next, we describe our experimental protocol, evaluation metrics and compared methods.

5.1 TRAINING

The policy is trained via PPO [Schulman et al., 2017]. The agent receives a positive reward whenever it retrieves the correct object. The agent receives no reward when it chooses the incorrect object and the episode ends. To solve the task as quickly as possible, the agent receives a negative slack penalty. We used a positive reward of 2.0, and a slack penalty of -0.1 for our experiments.

5.2 EXPERIMENTAL PROTOCOL

Shapes and Colors. For visual objects, we use 50 randomly sampled classes from CIFAR-100 [Krizhevsky, 2009] and obtain icons from NounProject (`https://thenounproject.com/`), a dataset of black-and-white

icons. Our dataset consists of 60 shapes like {*fish, bike, spider, castle*}, which we color using 13 colors {*red, green, blue, cyan, yellow, grey, orange, purple, brown, pink, violet, puff, olive*}. From these 60 shapes and 13 colors, we randomly choose 50 shapes and 10 colors as seen concepts. The remaining 10 shapes and 3 colors are designated as unseen novel concepts, used for evaluation at test time.

Held-out Set for Policy Training. If the agent would have been trained with all the shapes and colors available in the training set, it would have never encountered a situation with a novel concept, which is the main goal of this paper. To expose the agent to scenes with novel concepts, we held-out a subset of 10 shapes and 3 colors from the seen set of 50 seen shapes and 10 seen color. We treat these held-out concepts as unseen during training stage. We do not expose the agent to the unseen test set during the training time. Thus, when learning the policy, we do not pre-train the perception to recognize the remaining 40 shapes and 7 colors. For the single-object task, the training dataset for policy training consists of 1000 episodes in which the instruction contains a seen object, as well as, 1000 episodes which contain novel objects in the instruction. For the two-object task, the training dataset consists of 1000 episodes. The unseen attributes mentioned in the instruction in both these datasets are chosen from the held-out subset (10 shapes, 3 colors).

Test Set. At test time, we introduce 10 new shapes and 3 new colors. These 10 shapes and 3 colors are different from the concepts in the held-out subset and were never observed before. For the *single-object* task, the test-set contains 100 episodes where the instruction refers to a seen object and 100 episodes instructions referring to an unseen one. For the *two-object* task, the test-set contains 100 episodes whose instructions refer to two novel objects (as in Figure 1b).

Evaluation Metrics. We evaluate the compared approaches by the accuracy in choosing the objects in the instruction. For the single-object task, we report 'Seen Success (%)' which calculates the accuracy of agent in picking the right object when the instruction mentions a seen concept. Additionally, we report 'Unseen Success (%)', which calculates the accuracy of the agent in picking the right novel object when the instruction mentions an unseen object (either single or both attributes are unknown. For the two-object task, we report '1$^{\text{st}}$ Object Success (%)' which computes how frequently the agent correctly infers at least one of the objects mentioned in the instruction. We also report '2$^{\text{nd}}$ Object Success (%)' which computes how frequently the agent correctly retrieves both the objects mentioned in the instruction.

5.3 COMPARED METHODS

Single-Object Task: We compare with the following:

1. **Random**: Pick one of the $|\mathcal{O}|$ objects in the room uniformly at random.
2. **Raw Features**: It is expected that using raw features would work well for seen concepts, but not for novel categories since the perception model was not trained on unseen concepts. This policy takes just these raw features as input and and does not explicitly generate the internal state representation.
3. **Without Top-1 Score and Instruction Embedding**: In this ablation, we only use the similarity score $\phi^S(o, i)$ and $\phi^C(o, i)$. We do not feed the top-1 scores and the instruction $2m$-dimensional instruction embedding. This will help us evaluate the importance of feeding top-1 scores along with the similarity scores.
4. **Without Instruction Embedding**: In this ablation, we use full internal state tuple but do not feed the instruction $2m$-dimensional instruction embedding.
5. **Our approach**: Our approach, which uses the full internal state tuple as well as the instruction embedding.

Two-object Task. We compare our method with the following baselines. All baselines utilize the full input: the internal state tuple, the agent's position, and the instruction embedding.

1. **Random**: Pick two objects of the $|\mathcal{O}|$ objects in the room uniformly at random.
2. **Without Gradient Update**: In this ablation, after the agent picks the first object, we will not update the perception model to recognize the novel concept. This will help us evaluate the importance of the one-shot learning step.
3. **With Gradient Update**: This is our approach. In this, after the agent infers one of the objects specified in the instruction, it updates the perception model and uses the updated perception module to infer the other object mentioned in the instruction.
4. **Oracle Perception**: To get an upper bound on the performance on this task using our policy, we also replace the perception model with an oracle perception model that was trained on *all seen and unseen concepts*.

6 RESULTS

6.1 SINGLE OBJECT TASK

Table 1 gives the accuracy of picking the correct object in the single-object task. Several observation worth noting. First, using similarity scores leads to better performance than using raw feature vectors. Second, using top-1 scores boosts reasoning performance, since it helps calibrate the similarity score for seen *vs.* unseen concept. Finally, using instruction embedding is necessary for reasoning. We found that even for seen instructions, the agent learns to stop at objects which only matches part of the instruction (*e.g.*, when the shape matches but not the color). Explicitly specifying which attributes of the instruction are seen vs. unseen helps

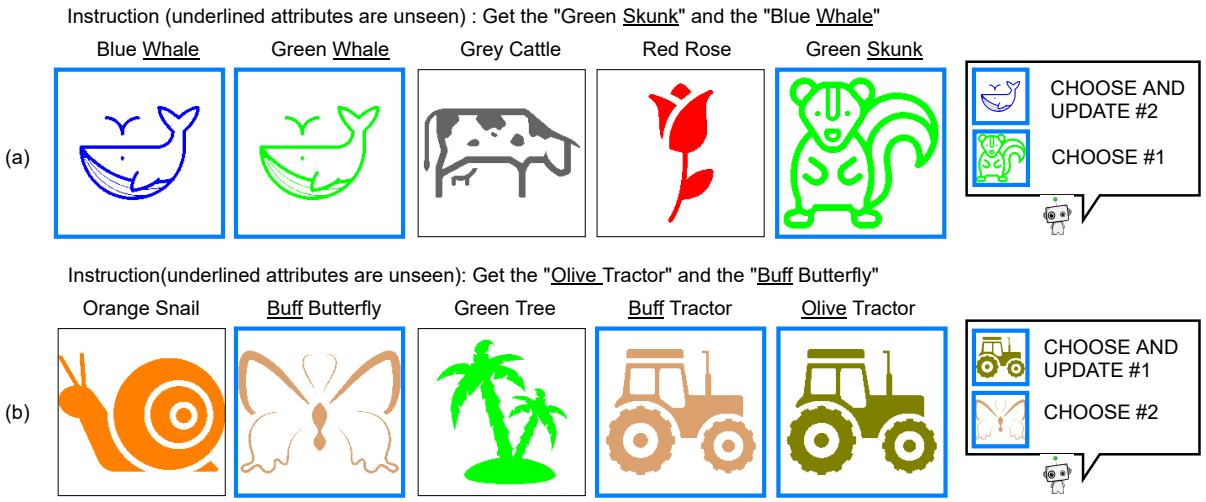


Figure 4: Two-Object Task Qualitative Examples. In the figure, underlined attributes are unseen. Object with blue boundaries are novel. In (a), the agent walks through the environment and infers the concept 'Whale' because there is no other blue-colored object. After updating its perception model, the agent is able to disambiguate between 'Whale' and 'Skunk' and chooses the 'Green Skunk' correctly. In (b), the agent can reason about both unseen colors using elimination.

#	Method	Sim Score	Top-1 Score	Ins. Type	Seen Success (%) ↑	Unseen Success (%) ↑
1	Random				$20_{\pm 3.97}$	$20_{\pm 3.97}$
2	Raw Features				$100_{\pm 0}$	$33_{\pm 2.2}$
3	Ours	✓			$88_{\pm 3.24}$	$67_{\pm 4.72}$
4	Ours	✓	✓		$87_{\pm 3.36}$	$79_{\pm 4.01}$
5	Ours	✓	✓	✓	$97_{\pm 1.40}$	$87_{\pm 3.36}$

Table 1: **Evaluation on the single object task**.

the agent to continue exploring and stopping at an object that fully matches the instruction.

6.2 TWO-OBJECT TASK

	Method	w/ Gradient Update	1st object Success (%) ↑	2nd object success (%) ↑
1	Random	-	$20_{\pm 0.129}$	$5_{\pm 0.07}$
2	Ours	-	$86_{\pm 3.85}$	$22_{\pm 4.15}$
3	Ours	✓	$82_{\pm 3.84}$	$33_{\pm 4.70}$
4	Oracle Perception	-	$95_{\pm 2.18}$	$95_{\pm 2.18}$

Table 2: **Evaluation on the two unseen object task**

Table 2 gives the accuracy in the two-object task, leading to the following observations. First, picking the first object using our perception and reasoning module is easy. The agent can often successfully reasons about one of the objects that is unambiguous. Second, the agent performs poorly without a 1-shot learning step to learn the unseen concept associated with the first object. In this case, it cannot disambiguate between two objects in the scene that can potentially match with the second object in the instruction. Third, when the agent updates its perception module with a 1-shot learning step, it better resolves the ambiguity between the two objects. We provide qualitative examples in Figure 4.

To showcase the efficacy of the policy, we also run an experiment with an oracle perception model that was trained using both seen and unseen concepts. This sets an upper bound on the performance of the policy as all concepts are considered seen, and therefore there is no ambiguity while reasoning about the two objects specified in the instruction.

7 CONCLUSIONS

This paper addressed the problem of training an agent to reason over a set of objects and learn to use elimination for learning new concepts in a zero-shot manner. We find that agents can be successfully trained using reinforcement learning to achieve this task. This study introduces a new approach to combining deep learning with abstract reasoning. Our framework for reasoning-by-elimination can be generalized to more natural visual scenarios because it does not depend on a specific perception module. The distribution of perception module scores, which are used as input features for the reasoning module, can reflect uncertainties about out-of-distribution samples in natural images, as demonstrated in Hendrycks and Gimpel [2017], Atzmon and Chechik [2019], Vyas et al. [2018]. While the current paper developed the ideas in a simple synthetic world, we show through ablations that every component of the proposed approach is necessary for success in this task. Various applications are likely to benefit from RBE. For example, RBE is commonly used to reach a diagnosis of exclusion in medicine

and is used for differential diagnosis. RBE is also important for fault diagnosis in computer systems and manufacturing. Future research directions include automated approaches to RBE with less feature engineering; finding how to collect such “RBE" labels at scale; and considering concepts from active learning for the RBE problem.

References

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, I. Reid, Stephen Gould, and A. V. D. Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3674–3683, 2018.

Jacob Andreas and D. Klein. Alignment-based compositional semantics for instruction following. In *Conference on Empirical Methods in Natural Language Processing, EMNLP*, 2015.

Yuval Atzmon and Gal Chechik. Probabilistic and-or attribute grouping for zero-shot learning. In *Proceedings of the Thirty-Forth Conference on Uncertainty in Artificial Intelligence*, 2018.

Yuval Atzmon and Gal Chechik. Adaptive confidence smoothing for generalized zero-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11671–11680, 2019.

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

S. Bach, M. Broecheler, B. Huang, and L. Getoor. Hinge-loss markov random fields and probabilistic soft logic. *Journal of Machine Learning Research*, 2017.

Valts Blukis, Yannick Terme, Eyvind Niklasson, Ross A. Knepper, and Yoav Artzi. Learning to map natural language instructions to physical quadcopter control using simulated flight. In *CoRL*, 2019.

Valts Blukis, Ross A. Knepper, and Yoav Artzi. Few-shot object grounding and mapping for natural language robot instruction following. *ArXiv*, abs/2011.07384, 2020.

Nicolò Cesana-Arlotti, Ágnes Melinda Kovács, and Ernő Téglás. Infants recruit logic to learn about the social world. *Nature communications*, 11(1):1–9, 2020.

David L. Chen and R. Mooney. Learning to interpret natural language navigation instructions from observations. In *AAAI 2011*, 2011.

Arthur Conan Doyle. *The Sign of Four*. Lippincott's Monthly Magazine, 1932.

J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

Felix Duvallet, T. Kollar, and A. Stentz. Imitation learning for natural language direction following through unknown environments. *2013 IEEE International Conference on Robotics and Automation*, pages 1047–1053, 2013.

Claudia Fugazza, Attila Andics, Lilla Magyari, Shany Dror, András Zempléni, and Ádám Miklósi. Rapid learning of object names in dogs. *Scientific reports*, 11(1), 2021.

Myrto Grigoroglou, Sharon Chan, and Patricia A Ganea. Toddlers' understanding and use of verbal negation in inferential reasoning search tasks. *Journal of experimental child psychology*, 183:222–241, 2019.

Justin Halberda. Is this a dax which i see before me? use of the logical argument disjunctive syllogism supports word-learning in children and adults. *Cognitive psychology*, 53 (4):310–344, 2006.

S. Hemachandra, Felix Duvallet, T. M. Howard, N. Roy, A. Stentz, and Matthew R. Walter. Learning models for following natural language directions in unknown environments. *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5608–5615, 2015.

D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017.

Felix Hill, Stephen Clark, Karl Moritz Hermann, and Phil Blunsom. Understanding early word learning in situated artificial agents. *arXiv preprint arXiv:1710.09867*, 2017.

Felix Hill, Olivier Tieleman, Tamara von Glehn, Nathaniel Wong, Hamza Merzic, and Stephen Clark. Grounded language learning fast and slow. In *International Conference on Learning Representations*, 2021.

Patrick J Hurley. *A concise introduction to logic*. Nelson Education, 2014.

M. Jänner, Karthik Narasimhan, and R. Barzilay. Representation learning for grounded spatial reasoning. *Transactions of the Association for Computational Linguistics*, 6: 49–61, 2018.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017.

A. Kimmig, S. Bach, M. Broecheler, B. Huang, and L. Getoor. A short introduction to probabilistic soft logic. In *NIPS Workshop on Probabilistic Programming*, 2012.

Jacob Krantz, Erik Wijmans, A. Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. *ArXiv*, abs/2004.02857, 2020.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

C.H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009.

Jelena Luketina, Nantas Nardelli, Gregory Farquhar, Jakob N. Foerster, Jacob Andreas, Edward Grefenstette, S. Whiteson, and Tim Rocktäschel. A survey of reinforcement learning informed by natural language. In *IJCAI*, 2019.

Massimiliano Mancini, Muhammad Ferjad Naeem, Yongqin Xian, and Zeynep Akata. Open world compositional zero-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021.

Ellen M Markman. Constraints children place on word meanings. *Cognitive science*, 14(1):57–77, 1990.

Ellen M Markman, Judith L Wasow, and Mikkel B Hansen. Use of the mutual exclusivity assumption by young word learners. *Cognitive psychology*, 47(3):241–275, 2003.

Dipendra Kumar Misra, J. Langford, and Yoav Artzi. Mapping instructions and visual observations to actions with reinforcement learning. In *Conference on Empirical Methods in Natural Language Processing, EMNLP*, 2017a.

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*, pages 1792–1801, 2017b.

Shilpa Mody and Susan Carey. The emergence of reasoning by the disjunctive syllogism in early childhood. *Cognition*, 154:40–48, 2016.

MF Naeem, Y Xian, F Tombari, and Zeynep Akata. Learning graph embeddings for compositional zero-shot learning. In *34th IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2021.

Tushar Nagarajan and Kristen Grauman. Attributes as operators: factorizing unseen attribute-object compositions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 169–185, 2018.

Senthil Purushwalkam, Maximilian Nickel, Abhinav Gupta, and Marc'Aurelio Ranzato. Task-driven modular networks for zero-shot compositional learning. In *ICCV*, 2019.

Yuankai Qi, Qi Wu, P. Anderson, Xin Wang, W. Wang, Chunhua Shen, and A. V. D. Hengel. Reverie: Remote embodied visual referring expression in real indoor environments. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9979–9988, 2020.

H. Reckman, Jeff Orkin, and D. Roy. Learning meanings of words and constructions, grounded in a virtual game. In *KONVENS*, 2010.

M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1), 2006.

B. Romera-Paredes and P. Torr. An embarrassingly simple approach to zero-shot learning. In *ICML*, 2015.

John Schulman, F. Wolski, Prafulla Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.

Stefanie Tellex, T. Kollar, Steven Dickerson, Matthew R. Walter, A. Banerjee, S. Teller, and N. Roy. Approaching the symbol grounding problem with probabilistic graphical models. *AI Mag.*, 32:64–76, 2011.

A. Vyas, N. Jammalamadaka, X. Zhu, S. Das, B. Kaul, and T. L. Willke. Out-of-distribution detection using an ensemble of self supervised leave-out classifiers. In *ECCV*, 2018.

Hai Wang and Hoifung Poon. Deep probabilistic logic: A unifying framework for indirect supervision. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 1891–1902, 2018.

Y. Xian, B. Schiele, and Z. Akata. Zero-shot learning - the good, the bad and the ugly. In *CVPR*, 2017.

Fei Xu and Joshua B Tenenbaum. Word learning as bayesian inference. *Psychological review*, 114(2):245, 2007.

Yuhao Zhang, Hang Jiang, Y. Miura, Christopher D. Manning, and C. Langlotz. Contrastive learning of medical visual representations from paired images and text. *ArXiv*, abs/2010.00747, 2020.