GROUNDED COMPOSITIONAL GENERALIZATION WITH ENVIRONMENT INTERACTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper, we present a compositional generalization approach in grounded agent instruction learning. Compositional generalization is important part of human intelligence, but current neural network models do not have such ability. This is more complicated in multi-modal problems with grounding. Our proposed approach has two main ideas. First, we use interactions between agent and the environment to find components in the output. Second, we apply entropy regularization with common network architectures to learn corresponding input components for each output component. The result shows the proposed algorithm has significantly higher performances than baselines in most tasks, with more than 25% absolute average accuracy increase. We also show the impact of entropy regularization and other changes with ablation study. We hope this work is the first step to address grounded compositional generalization, and it will be helpful in advancing artificial intelligence research. The source code is included in supplementary material.

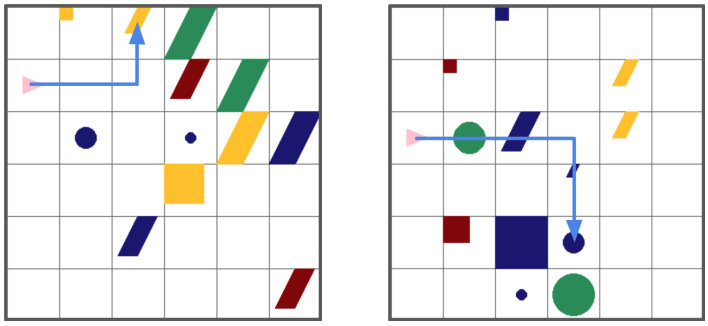
1 INTRODUCTION

Compositional generalization is a key skill for flexible and efficient learning. Humans leverage compositionality to create and recognize new combinations of familiar concepts (Chomsky, 1957; Minsky, 1986). Though there are many progresses for machine learning and deep learning in various areas recently (LeCun et al., 2015), current main learning algorithms are not able to perform compositional generalization, and require many samples to train models. Such efficient learning is even more important when machines interact with the environment for grounding, because interactions are usually slow.

Machine learning has been mostly developed with an assumption that training and test distributions are identical. Compositional generalization, however, is a kind of out-of-distribution generalization (Bengio, 2017), where training and test distributions are different. During training, dataset does not contain the information of the difference, so it can only be given as prior knowledge. In compositional generalization, a sample is a combination of several components. Test distribution changes as test samples are new combinations of seen components in training. For example, if we can find "large apple" and "small orange" in some environments, then we can also find "large orange" among many objects in a new environment.

The recombination is enabled when an output component depends only on the corresponding input components, and invariant of other components (please see Section 4.1 for more details). So there are two aspects to consider. What are the components in output, and how to find the corresponding input signals. We propose to use interactions between agent and the environment to define output components. This simplification is analogues to model-free reinforcement learning (Sutton & Barto, 2018), where an agent does not have an environment model, but leans to act at each step during the interactions with the environment. Then we use entropy regularization (Li et al., 2019; Li & Eisner, 2019) to learn the minimal input components for outputs.

We evaluate the approach with gSCAN dataset (Ruis et al., 2020), which is designed to study compositional generalization in grounded agent instruction learning. Please see Figure 1 for examples. The results show the proposed approach has significantly higher performances than baselines in most tasks, with more than 25% absolute average accuracy increase. We also show the impact of



(a) Walk to a small cylinder. (b) Walk to a large blue circle.

Figure 1: Examples of gSCAN dataset. The dataset evaluates compositional generalization ability in grounded instruction learning. The agent needs to understand command and the environment to take sequence of actions. Training and test data have different distributions and tasks require different kinds of compositional generalization. Please refer to Section 5 for more details.

entropy regularization and other changes with ablation study. We hope this work will be helpful in advancing grounded compositional generalization and artificial intelligence research.

The contributions of this paper can be summarized as follows.

- This is the first work to enable accurate compositional generalization in grounded instruction learning problem.
- We find the combination of environment interaction and entropy regularization helps the generalization.

2 Related work

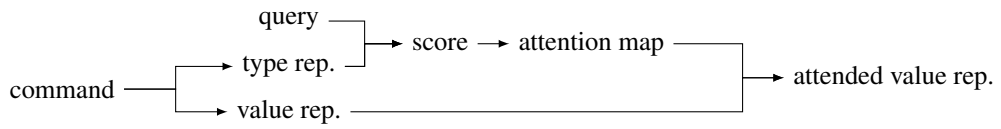
Compositional generalization research has a long history, and recently there are increasing focus on this area. SCAN dataset (Lake, 2019) was proposed to study compositional generalization in instruction learning. It maps a command sentence to a sequence of actions. This dataset has a property that input words and output actions have direct correspondence. Though some NLP tasks, such as machine translation, have similar property, not all problems fit to the setting. Also this dataset does not contain an environment for an agent to take actions.

SCAN dataset inspired multiple approaches (Russin et al., 2019; Andreas, 2019; Li et al., 2019; Lake, 2019; Gordon et al., 2020; Liu et al., 2020). Some of them lead to general techniques for compositional generalization. For example, entropy regularization (Li et al., 2019) is proposed to avoid redundant dependency on input, and it is a core idea of the approach in this paper.

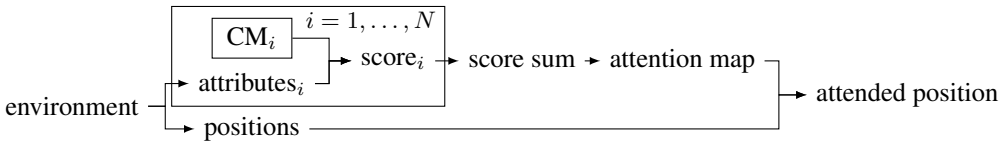
Compositional generalization has applications in various fields such as question answering (Andreas et al., 2016; Hudson & Manning, 2019; Keysers et al., 2020), counting (Rodriguez & Wiles, 1998; Weiss et al., 2018), systematic behaviour (Wong & Wang, 2007; Brakel & Frank, 2009), and hierarchical structure (Linzen et al., 2016). Another related work is independent disentangled representation (Higgins et al., 2017; Locatello et al., 2019), but they do not address compositional generalization. Compositionality is also helpful for reasoning (Talmor et al., 2020) and continual learning (Jin et al., 2020; Li et al., 2020).

Grounded SCAN (gSCAN) dataset was proposed to introduce environment to agent instruction learning with compositional generalization. It has a command sentence as input and a sequence of actions as output. However, the input command does not tell the specific way to act, but agent needs to understand the environment and take corresponding actions. This also avoids direct mapping between input words and output actions.

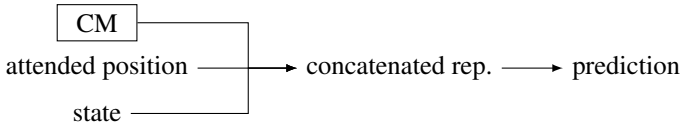
Different approaches have been proposed to address this problem. As compositional generalization requires prior knowledge for distribution change, these approaches correspond to different ways to provide the prior knowledge. Andreas (2019) uses linguistic knowledge to augment training data.



(a) Command module.



(b) Grounding module.



(c) Prediction module.

Figure 2: Flow charts for each module in the proposed approach. Command module and grounding module are based on attention mechanism as the main architecture, and prediction module concatenates inputs and use feed-forward network. More details can be found in Algorithm 1.

Kuo et al. (2020) uses external syntactic parser and WordNet. In this paper, we apply the prior knowledge for the interactions of agent and the environment.

3 PROBLEM DESCRIPTION

gSCAN dataset contains episodes of commands and actions, and each episode has an input and an output. The input includes a command and an environment. A command is a sequence of words. An environment contains an agent and a set of objects. An agent has its position and initial direction. An object has its position and attributes of color, shape and size. The output contains a sequence of actions (Figure 1).

We hope to extract prior knowledge from environment interactions. We first notice that an agent should know whether it is going to change position. This means we can break an episode to a sequence of steps, where each step corresponds to a position change. An agent should also know the change of directions between steps. So we separate the direction change and the adverb manner in each step. Therefore, we have three output for each step: direction, action and manner. We further convert the environment to be agent centered, and rotate the environment to make agent facing forward. We also assume automatic collision prevention. If the agent tries to push or pull an object to collide with other objects or boarder, then it stops. This makes us focus on addressing grounded compositional generalization problem.

In summary, when we consider agent interaction with the environment, we can convert the problem to a set of step-wise label prediction problems with multiple outputs. Input contains command, environment, state, and output contains direction, action and manner.

4 APPROACH

In this section, we describe the approach for grounded compositional generalization. As we use entropy regularization in different modules, we first introduce it, then move to the model architecture.

Algorithm 1 The proposed approach. Input includes command x^c, state s, object attributes x^a and object positions x^p. K is an embedding size. Please see Section 4 for more information.

Command module
Input: x^c, K
Output: r: embedding
$t = \text{EntReg}(E_t x^c)$
$v = \text{EntReg}(E_v x^c)$
$a^c = \text{Softmax}(qt/\tau^c)$
$u = va^c \in \mathbb{R}^{k_v}$
$r = \text{ff}_\theta(u, K) \in \mathbb{R}^K$

Grounding module
Input: x^c, x^a, x^p
Output: p: target position
for $i = 1, \ldots, N$ **do**
  $r_i = \text{CM}(x^c, V^a)$
  $x_i'^a = \text{ERL}(x^a)$
  $w_i = r_i x_i'^a \in \mathbb{R}^m$
end for
$w = \sum_{i=1}^N w_i$
$a^g = \text{Softmax}(w/\tau^g)$
$p = x^p a^g \in \mathbb{R}^{V^p}$

Prediction module
Input: x^c, x^a, x^p, s, K
Output: l: output logits
$p = \text{GM}(x^c, x^a, x^p)$
for each output type i **do**
  $k_i = \text{CM}(x^c, K) \in \mathbb{R}^K$
  for each node j **do**
    $p'_{i,j} = \text{ERL}(p)$
    $z_{i,j} = [k_i^T, p_{i,j}'^T, s^T]^T$
    $l_{i,j} = \text{ff}^\phi_{i,j}(z_{i,j}, 1)$
  end for
end for

4.1 Entropy regularization

The difficulty of compositional generalization is that there might be incorrect dependency between input and output components. For example, when "red" and "square" do not appear together in any training sample, a model might learn that square is not red. To avoid such case, we hope the representation of shape not influenced by input information of color.

Entropy regularization (Li et al., 2019; Li & Eisner, 2019) aims at reducing entropy of a representation, to avoid dependency on redundant input components. Given an representation x, we compute the L_2 norm and add normal noise to each element of the representation. In this way, the channel capacity decreases, so that the entropy for the representation reduces. Then, we feed the noised representation to the next layer, and add the norm to loss function.

$$\mathcal{L} = \mathcal{L}_{\text{original}} + \lambda L_2(x) \qquad \text{EntReg}(x) = x + \alpha \mathcal{N}(0, I)$$

where α is a weight for noise, positive for training and zero for inference.

A representation can be fed to multiple networks, requiring different regularization for each input node. So we design entropy regularization layer, where we achieve non-linear mapping by expanding each node x_i to a vector $h_i \in \mathbb{R}^H$ with ReLU activations, and maps it back to a node y_i, then apply entropy regularization to get y'_i. The input x and output $\text{ERL}(x) = y'$ have the same size.

$$h_i = \text{ff}_i^A(x_i, H), \qquad y_i = \text{ff}_i^B(h_i, 1), \qquad y' = \text{EntReg}(y).$$

We use EntReg for word embeddings and ERL for environment inputs.

4.2 Model architecture

Input contains one-hot representation of a command x^c with sentence length n and vocabulary size V^c, and a sequence of m objects. We denote object attributes x^a with vector size V^a, and object positions x^p with vector size V^p. We also has a binary state $s \in \{0, 1\}$ indicating whether this is the first step of an episode. Output has three types of classes. Direction y^d has C^d classes, action y^a has C^a classes, and manner y^m has C^m classes.

$$x^c = x_1^c, \ldots, x_n^c \in \mathbb{R}^{V^c \times n}, \quad x^a = x_1^a, \ldots, x_m^a \in \mathbb{R}^{V^a \times m}, \quad x^p = x_1^p, \ldots, x_m^p \in \mathbb{R}^{V^p \times m}.$$

The model has the following modules, as summarized in Figure 2 and Algorithm 1.

Command module (CM) This module takes one-hot representation of a command x^c as input and returns a representation r with size K, $r = \text{CM}(x^c, K)$. r is expected to be the embedding of certain type of keyword, e.g. action, color, etc. To enable compositional generalization, we hope to separate keys and values of attentions. For example, attention map of color word should depend only on the type of the word (color), but not on concrete value of the word (red or square). This avoids incorrect dependency (e.g. red color depends on square), so that the network generalizes to new combinations of values (e.g., green square). So we have each word x_i^c in command x^c has two embeddings for

type t_i and value v_i with corresponding embedding matrices, $E_t \in \mathbb{R}^{k_t \times V^c}, E_v \in \mathbb{R}^{k_v \times V^c}$, where k_t, k_v are embedding sizes, respectively. Similar to Li et al. (2019), we apply entropy regularization on both t and v to reduce redundant dependency.

$$t = \text{EntReg}(E_t x^c) \in \mathbb{R}^{k_t \times n}, \qquad v = \text{EntReg}(E_v x^c) \in \mathbb{R}^{k_v \times n}.$$

We use attention mechanism with a query $q \in \mathbb{R}^{k_t}$ as learnable parameters, keys t, values v, and temperature $\tau^c \in \mathbb{R}$. The attended value u is fed to a feed-forward network with output size K.

$$a^c = \text{Softmax}(qt/\tau^c) \in \mathbb{R}^n, \qquad u = va^c \in \mathbb{R}^{k_v}, \qquad r = \text{ff}_\theta(u, K) \in \mathbb{R}^K.$$

Grounding module (GM) This module finds target in the environment according to the command. We use N command modules for queries. For each query r_i, we have an dedicated entropy regularization layer on attributes $x_i'^a$, because different queries extract different type of attributes, and other attributes would be redundant. We then compute a score w_i. For $i = 1, \ldots, N$,

$$r_i = \text{CM}(x^c, V^a) \in \mathbb{R}^{V^a} \qquad x_i'^a = \text{ERL}(x^a) \in \mathbb{R}^{V^a \times m}, \qquad w_i = r_i x_i'^a \in \mathbb{R}^m.$$

The scores are added as w. Attention map a^g is computed by Softmax with temperature $\tau^g \in \mathbb{R}$, and is applied to get the attended object position p.

$$w = \sum_{i=1}^{N} w_i \in \mathbb{R}^m, \qquad a^g = \text{Softmax}(w/\tau^g) \in \mathbb{R}^m, \qquad p = x^p a^g \in \mathbb{R}^{V^p}.$$

Prediction module (PM) Prediction module takes command, environment and state as input, and outputs a prediction. We have three separate prediction modules for direction, action and manner, respectively. For each prediction module, we use one command module to extract a keyword k with size K for the prediction. We also have a environment module for target object position p, and all the prediction modules share it as an input. There is a input of state s, and this is also a shared input for each prediction module.

We observe that different output nodes are dependent on different input nodes. So we hope to reduce dependency of each output node to input nodes. We build a dedicated feed-forward network from input to each output node without weight sharing.

In each network of output type i and node j, we use entropy regularization layer for target position p, because different output nodes may need to be computed from different target directions, and other directions are redundant. Then all the inputs are concatenated to form a long input vector z, with size $L = K + V^p + 1$.

$$k_i = \text{CM}(x^c, K) \in \mathbb{R}^K, \qquad p'_{i,j} = \text{ERL}(p) \in \mathbb{R}^{V^p}, \qquad z_{i,j} = [k_i^T, p_{i,j}'^T, s^T]^T \in \mathbb{R}^L.$$

We then feed it to another feed-forward network to get a output node l_j. They are concatenated to form a logit l, and we use Softmax to output $\hat{y}$. C_i is the number of classes for the output type.

$$l_{i,j} = \text{ff}^\phi_{i,j}(z_{i,j}, 1) \in \mathbb{R}, \qquad l_i = [l_{i,1}, \ldots, l_{i,C_i}] \in \mathbb{R}^{C_i}, \qquad \hat{y}_i = \text{Softmax}(l_i) \in \mathbb{R}^{C_i}$$

We use cross entropy and the norms for entropy regularization with weight λ as training objective.

5 EXPERIMENTS

We use all eight tasks in gSCAN for experiment. Figure 1 shows examples of two gSCAN commands in different environments. Note that the main purpose of the experiment is not to compare with previous approaches, but to show that the designed compositional generalization problems are correctly addressed. The tasks are summarized as follows.

A: *Random* is from the same distribution as training data. It does not contain compositional generalization, and is used to compare with other tasks as reference.

B: *Yellow squares* has a new way of referring to a type of object, e.g., "small square" in training, but "yellow square" only in test.

C: *Red squares* has a new combination of attribute and object, e.g., "red" and "square" appears separately in training, but "red square" appears only in test.

D: *Novel direction* has target object in a new direction, e.g., down left direction only in test.

E: *Relativity* newly has objects of specific sizes as "small" or "large". e.g., refer size 2 as "large" (a sample does not contain larger sizes) only in test.

F: *Class inference* requires inferring object properties, e.g., the object size decides the number of pull or push actions to take.

G: *Adverbs* requires learning adverbs, e.g., "cautiously", with one or a few samples.

H: *Adverb to verb* has new combination of adverb and verb, e.g., "walking" and "while spinning" both appear in training, but "walking while spinning" appears only in test.

The hyper parameters and settings of the model and training, and more details of the dataset can be found in Appendix A. We evaluate accuracy on episodes. An episode is correct when all the steps are correct. Following Ruis et al. (2020), we run experiment three times, and report means and standard deviations.

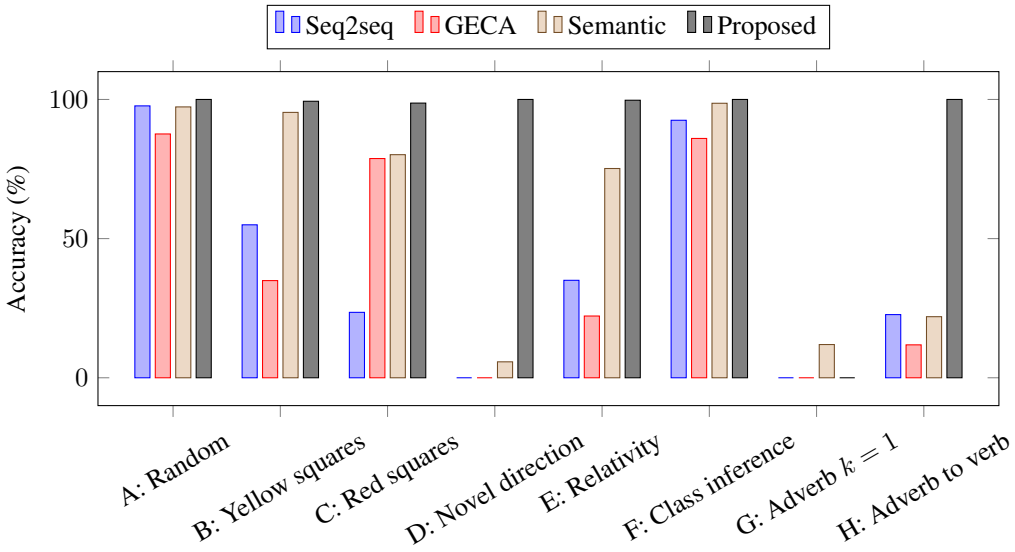


Figure 3: Bar graph of accuracy results. Seq2seq and GECA are from Ruis et al. (2020). Semantic is from Kuo et al. (2020). The results shows that the proposed approach can effectively address most of the tasks, and task G is analyzed in discussion section.

Table 1: Details of accuracy results (%) for experiments. We report mean and standard deviations.

	Seq2seq	**GECA**	**Semantic**	**Proposed**
A: Random	97.69 ± 0.22	87.60 ± 1.19	97.32	100.00 ± 0.00
B: Yellow squares	54.96 ± 39.39	34.92 ± 39.30	95.35	99.35 ± 0.84
C: Red squares	23.51 ± 21.82	78.77 ± 6.63	80.16	98.69 ± 1.85
D: Novel direction	0.00 ± 0.00	0.00 ± 0.00	5.73	100.00 ± 0.00
E: Relativity	35.02 ± 2.35	33.19 ± 3.69	75.19	99.72 ± 0.38
F: Class inference	92.52 ± 6.75	85.99 ± 0.85	98.63	100.00 ± 0.00
G: Adverb $k = 1$	0.00 ± 0.00	0.00 ± 0.00	11.94	0.00 ± 0.00
H: Adverb to verb	22.70 ± 4.59	11.83 ± 0.31	21.95	100.00 ± 0.00

The results are shown in Figure 3 and more details in Table 1. The proposed approach has significantly higher accuracy than the baseline methods in most of the tasks. It does not work in Experiment G, and we discuss about it in Section 6.3. This indicates that the approach can effectively address most of the designed compositional generalization problems. Especially in Experiment D:

Novel direction and Experiment H: Adverb to verb, there are more than 94% and 78% absolute improvement, respectively. Experiment D requires recombining different directions, and experiment H requires recombining verbs and adverbs. These tasks are classical compositional generalization problems with recombination of seen components.

6 DISCUSSIONS

In this section, we discuss and analyze more details to have better understanding of the approach. We will first run ablation study to find what factors enable the improvement. We then analyze the grounding ability of the proposed approach. We also discuss the one-shot learning problem. We further apply the approach on target length task.

6.1 ABLATION STUDY

We run ablation study to understand what factors of the algorithm are important. One change of the approach is entropy regularization (EntReg). We also have structure change to apply the regularization (ERL). We have element-wise prediction for each output node (NodeOut). We design experiments by removing each of these factors.

The results (Table 2) show that the ablation experiments have lower accuracy than the proposed approach. This means all the changes are important for the approach. The reduction is especially significant for Experiment D: Novel direction. In this task, each value of the output depends on different input components, because different output values have priorities in the setting. When target appears on back, back right or back left, the agent turns around. When it appears on right, the agent turns right. Setting agent faces horizontal positive direction, this means turning around node depends only on one coordinate (horizontal is negative), but turn right node depends on two coordinates (horizontal is zero and vertical is negative). This is different from previous compositional generalization where all values of an output component depend on the same input component. Though this particular case may be solved in other ways, it is general that different output values may depend on different input components, and the proposed method is able to address the problem.

Table 2: Accuracy results (%) for ablation study. The ablation experiments have lower accuracy than the proposed approach. This means all the changes are important for the approach.

	Proposed	**No EntReg**	**No ERL**	**No NodeOut**
A: Random	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	95.51 ± 6.35
B: Yellow squares	99.35 ± 0.84	94.60 ± 4.37	74.50 ± 0.57	95.73 ± 5.20
C: Red squares	98.69 ± 1.85	98.27 ± 1.38	94.72 ± 2.96	96.40 ± 5.09
D: Novel direction	100.00 ± 0.00	92.44 ± 7.03	13.29 ± 14.63	0.00 ± 0.00
E: Relativity	99.72 ± 0.38	99.69 ± 0.43	96.57 ± 3.17	95.98 ± 5.67
F: Class inference	100.00 ± 0.00	100.00 ± 0.00	99.99 ± 0.01	95.76 ± 6.00
G: Adverb $k = 1$	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
H: Adverb to verb	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	96.67 ± 4.71

6.2 GROUNDING

We run visualization to show that the approach works in the expected way. The grounding finds correct object according to descriptions in command. A common way for grounding, which is also used in the proposed approach, is to use attention. So we evaluate whether the correct target object is attended.

The results are shown in Table 3. This indicates that the proposed approach has good grounding accuracy in all the tasks. Note that it also works on Experiment G, where the final prediction is wrong. This means the errors in Experiment G are not caused by grounding, and the proposed approach addresses the designed grounding problems in all the tasks of gSCAN.

Table 3: Grounding accuracy results (%) for the proposed method. Grounding works well in all the tasks.

A: Random	100.00 ± 0.00	E: Relativity	100.00 ± 0.00
B: Yellow squares	100.00 ± 0.00	F: Class inference	100.00 ± 0.00
C: Red squares	98.91 ± 1.54	G: Adverb $k = 1$	98.68 ± 0.88
D: Novel direction	100.00 ± 0.00	H: Adverb to verb	100.00 ± 0.00

6.3 ONE-SHOT LEARNING

The gSCAN experiment result shows that Experiment G: Adverb does not work for methods without external lexical information. This experiment was designed as few-shot or one-shot ($k = 1$) learning task, however it is different from standard settings. Conventionally, one-shot learning has only one sample for a class, but the frequency or weight of the sample is not fixed. For example, in Lake & Baroni (2018), the one-shot sample is repeated to be 10% of the whole dataset. Changing the weight of one-shot samples is also common in human learning, as we focus on one special sample and learn from it. However, in experiment G, the frequency of the one-shot sample is one, while there are more 300,000 other training samples.

This not only extremely lower the contribution of the sample to prediction loss, but also decrease the chance to be included in a mini-batch for training. To some extent, this requires the algorithm to partially address catastrophic forgetting (Kirkpatrick et al., 2017). This is because when the last mini-batches do not contain the sample, the model should not forget it. So this experiment contains more challenges than core compositional generalization, and they are beyond the scope of this paper.

6.4 TARGET LENGTH TASK

Ruis et al. (2020) also contains a separate dataset for target length task. It requires the model to perform on the target action sequence longer than those in training. This is not the focus of this paper, but we run experiments for completeness.

The result is in Table 4. It shows that, for the proposed approach, the accuracy does not drastically decrease when the evaluation target length increases. This might be a because of the bi-product of step-wise prediction, which is less influenced by the length of an episode. The result indicates an additional benefit of using environment interactions.

Table 4: Accuracy results (%) for target length experiments. Seq2seq is from Ruis et al. (2020). Semantic is from Kuo et al. (2020). The proposed approach avoids drastic decrease when the length increases.

Target length	**Seq2seq**	**Semantic**	**Proposed**
15	94.98 ± 0.12	93.43	94.45 ± 4.56
16	19.32 ± 0.02	90.88	96.28 ± 3.36
17	1.71 ± 0.38	87.92	96.04 ± 3.25
$\geq$18	< 1.00	56.50	97.64 ± 2.06

7 CONCLUSION

We propose an approach to address compositional generalization in grounded the agent instruction learning. The approach uses interactions between agent and the environment to define output components, and entropy regularization to reduce redundant dependency on input. We achieve significant improvements in most of gSCAN tasks, indicating that it addresses the designed grounded compositional generalization problems in these tasks. In ablation study, we show the effectiveness of entropy regularization and other changes, and look into different aspects of the approach. We hope this work will be a step towards understanding compositional generalization in grounded language learning and general artificial intelligence.

REFERENCES

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016.

Jacob Andreas. Good-enough compositional data augmentation. *arXiv preprint arXiv:1904.09545*, 2019.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Yoshua Bengio. The consciousness prior. *arXiv preprint arXiv:1709.08568*, 2017.

Philémon Brakel and Stefan Frank. Strong systematicity in sentence processing by simple recurrent networks. In *31th Annual Conference of the Cognitive Science Society (COGSCI-2009)*, pp. 1599–1604. Cognitive Science Society, 2009.

Noam Chomsky. *Syntactic structures*. Walter de Gruyter, 1957.

Jonathan Gordon, David Lopez-Paz, Marco Baroni, and Diane Bouchacourt. Permutation equivariant models for compositional generalization in language. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=SylVNerFvr`.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations (ICLR)*, 2017.

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.

Xisen Jin, Junyi Du, and Xiang Ren. Visually grounded continual learning of compositional semantics. *arXiv preprint arXiv:2005.00785*, 2020.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. Measuring compositional generalization: A comprehensive method on realistic data. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=SygcCnNKwr`.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Yen-Ling Kuo, Boris Katz, and Andrei Barbu. Compositional networks enable systematic generalization for grounded language understanding. *arXiv preprint arXiv:2008.02742*, 2020.

Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International Conference on Machine Learning*, pp. 2873–2882, 2018.

Brenden M Lake. Compositional generalization through meta sequence-to-sequence learning. In *Advances in Neural Information Processing Systems*, pp. 9788–9798, 2019.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

Xiang Lisa Li and Jason Eisner. Specializing word embeddings (for parsing) by information bottleneck. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing*, pp. 2744–2754, Hong Kong, November 2019. URL http://cs.jhu.edu/~jason/papers/#li-eisner-2019. Best Paper Award.

Yuanpeng Li, Liang Zhao, Jianyu Wang, and Joel Hestness. Compositional generalization for primitive substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4284–4293, 2019.

Yuanpeng Li, Liang Zhao, Kenneth Church, and Mohamed Elhoseiny. Compositional continual language learning. In *International Conference on Learning Representations*, 2020.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. Assessing the ability of lstms to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535, 2016.

Qian Liu, Shengnan An, Jian-Guang Lou, Bei Chen, Zeqi Lin, Yan Gao, Bin Zhou, Nanning Zheng, and Dongmei Zhang. Compositional generalization by learning analytical expressions. *arXiv preprint arXiv:2006.10627*, 2020.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning*, pp. 4114–4124, 2019.

Marvin Minsky. *Society of mind*. Simon and Schuster, 1986.

Paul Rodriguez and Janet Wiles. Recurrent neural networks can learn to implement symbol-sensitive counting. In *Advances in Neural Information Processing Systems*, pp. 87–93, 1998.

Laura Ruis, Jacob Andreas, Marco Baroni, Diane Bouchacourt, and Brenden M Lake. A benchmark for systematic generalization in grounded language understanding. *arXiv preprint arXiv:2003.05161*, 2020.

Jake Russin, Jason Jo, and Randall C O'Reilly. Compositional generalization in a deep seq2seq model by separating syntax and semantics. *arXiv preprint arXiv:1904.09708*, 2019.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Alon Talmor, Oyvind Tafjord, Peter Clark, Yoav Goldberg, and Jonathan Berant. Teaching pre-trained models to systematically reason over implicit knowledge. *arXiv preprint arXiv:2006.06609*, 2020.

Gail Weiss, Yoav Goldberg, and Eran Yahav. On the practical computational power of finite precision rnns for language recognition. *arXiv preprint arXiv:1805.04908*, 2018.

Francis CK Wong and William SY Wang. Generalisation towards combinatorial productivity in language acquisition by simple recurrent networks. In *2007 International Conference on Integration of Knowledge Intensive Multi-Agent Systems*, pp. 139–144. IEEE, 2007.

A EXPERIMENT SETTINGS

We implement the model as follows. In command module, we use $k_t = 32$ and $k_v = 32$. ff_θ is a one layer feed forward network with linear activation. τ^c = 1. In grounding module, we use $N = 3$ and $\tau^g = 0.1$. In prediction module, We use $K = 8$. ff_ϕ is a two layer feed forward network. The first layer has 32 hidden nodes with ReLU activation. The second layer has linear activation. We use $U = 32$. ff_ψ is a three layer feed forward network. The first and second layers have 32 hidden nodes with Relu activation. The third layer has one output node with linear activation. For target length task, the setting is the same except $N = 8$ in grounding module.

In training, we use $\lambda = 1$ and $\alpha = 0.1$. We use Adam optimizer (Kingma & Ba, 2014) with 10,000 training steps. Each mini-batch has 256 samples selected from training data uniformly at random with replacement. Gradient is clip by norm of 1. We use initial learning rate 0.001, which exponentially decays every 100 steps by a factor of 0.96. TensorFlow (Abadi et al., 2016) is used for implementation.

The sample size of gSCAN dataset is summarized in Table 5.

Table 5: Sample size of gSCAN dataset.

Task	Samples
Train	367,933
A: Random	19,282
B: Yellow squares	18,718
C: Red squares	37,436
D: Novel direction	88,642
E: Relativity	16,808
F: Class inference	11,460
G: Adverb $k = 1$	112,880
H: Adverb to verb	38,582

B FEW-SHOT LEARNING

The results of other few-shot learning in Experiment G are summarized in Table 6.

Table 6: Accuracy results (%) for few-shot learning experiments. Seq2seq is from Ruis et al. (2020). Semantic is from Kuo et al. (2020).

	Seq2seq	**Semantic**	**Proposed**
Adverb $k = 1$	0.00 ± 0.00	11.94	0.00 ± 0.00
Adverb $k = 5$	0.47 ± 0.14	10.17	0.00 ± 0.00
Adverb $k = 10$	2.04 ± 0.95	33.28	0.00 ± 0.00
Adverb $k = 15$	-	40.78	0.00 ± 0.00
Adverb $k = 50$	4.63 ± 2.08	-	67.44 ± 28.27