Compositional Data and Task Augmentation for Instruction Following

Soham Dan[1]*, **Xinran Han**[2]*, **and Dan Roth**[1]
[1]University of Pennsylvania
[2]Harvard University
{sohamdan,hxinran,danroth}@seas.upenn.edu

Abstract

Executing natural language instructions in a physically grounded domain requires a model that understands both spatial concepts such as *left_of* and *above*, and the compositional language used to identify landmarks and articulate instructions relative to them. In this paper, we study instruction understanding in the blocks world domain. Given an initial arrangement of blocks and a natural language instruction, the system executes the instruction by manipulating selected blocks. The highly compositional instructions are composed of atomic components and understanding these components is a necessary step to executing the instruction. We show that while end-to-end training (supervised only by the correct block location) fails to address the challenges of this task and performs poorly on instructions involving a single atomic component, knowledge-free auxiliary signals can be used to significantly improve performance by providing supervision for the instruction's components. Specifically, we generate signals that aim at helping the model gradually understand components of the compositional instructions, as well as those that help it better understand spatial concepts, and show their benefit to the overall task for two datasets and two state-of-the-art (SOTA) models, especially when the training data is limited—which is usual in such tasks.

1 Introduction

One of the hallmarks of artificial intelligence is designing robots that can understand and execute natural language instructions in a grounded domain (Winograd, 1972) . There is a strong need for this technology in several applications (Branavan et al., 2009; Tellex et al., 2011; Chen and Mooney, 2011), where the robot needs to ground relevant parts of the instruction to the environment. Blocks World

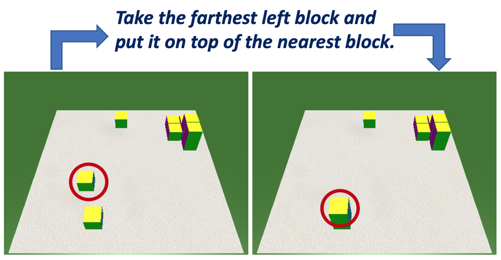
Take the farthest left block and put it on top of the nearest block.

Figure 1: Task: Given a configuration of blocks and an instruction, predict the source block and target location. Note the multiple spatial concepts and compositional nature of the instruction. Our proposed approach correctly identifies the source block while the SOTA (Tan and Bansal, 2018) model fails on this example.

is a popular platform to study instruction understanding in physically grounded environments and presents several key reasoning challenges (Winograd, 1972; Narayan-Chen et al., 2019; Jayannavar et al., 2020; Bisk et al., 2016; Mehta and Goldwasser, 2019; Tan and Bansal, 2018; Misra et al., 2017; Bisk et al., 2018). In Bisk et al. (2016), the environment consists of a number of blocks placed on a board. The robot receives an instruction and the current block configuration as input and has to execute the instruction by manipulating appropriate blocks. There are two scenarios: the easier one where blocks have labels and the more challenging one in which the blocks are unlabeled, thus necessitating the use of involved referential expressions. In the labeled dataset, the blocks have names: *move the nvidia block to the right of the hp block*, making grounding easier. However, as shown in Fig. 1, instructions in the unlabeled dataset are highly complex, involving multiple spatial concepts and a high degree of compositionality. To identify the block to be moved (source block) one needs to ground *the block nearest (to) the left hand corner* and understand the *above* and *right* spatial concepts. Similarly, identifying where the block is to be moved to (target location) requires grounding *the block below the centermost block* and understanding the

* The first two authors contributed equally. Work done while the second author was at the University of Pennsylvania.
http://cogcomp.org/page/publication_view/951

spatial concepts *below* and *right*. In this paper we ask a fundamental question: do models trained on small data-sets for a grounded task, really learn compositional reasoning or do they merely over-fit to a particular data-set ? We show that existing models trained in an end to end manner to predict source block and target coordinates, given the instruction and current configuration of blocks, fail to generalize to simple instructions. Existing approaches do not address the compositionality of these instructions nor deal directly with the complex spatial concepts. In this paper, we attempt to bridge this gap by augmenting the end-to-end training in a knowledge-free way, with (i) data augmentation and (ii) task augmentation, to improve performance on the standard test set. Further, we show in Sec 3.1, while existing models perform poorly on the generated atomic instructions, our approach removes this vulnerability.

In (i) we use a few simple templates to automatically generate examples that focus on a single spatial concept. Then, we pre-train the model on this synthetic data before training on the more complex, original training data. This approach is an instance of **Curriculum Learning** (Bengio et al., 2009) where the difficulty of an instance is related to the number of spatial concepts it contains. In (ii), we create auxiliary tasks which are coarser than the location prediction task (e.g. quadrant prediction) and train the model on these tasks jointly with the main task (Thung and Wee, 2018). The auxiliary tasks help teach the model spatial concepts by providing explicit supervision for these components in the instruction. We supervise the auxiliary tasks in an alternate fashion with the main task to train the model. We emphasize that *both our proposed solutions require no additional supervision*. We observe that compositional data augmentation and the auxiliary tasks improve generalization on the synthetic test data and on the standard test data. Our method is evaluated on different datasets (labeled blocks and unlabeled blocks) and on different models (Bisk et al., 2016; Tan and Bansal, 2018).

2 Augmenting End-to-End Training

Given the current configuration ((x, y, z) locations of a maximum of 20 blocks) and an instruction, the model has to move the corresponding block. This has two sub-tasks—(1) Source Prediction: predict the block to be moved and (2) Target Prediction: predict the location the source block is to be moved to. Figure 2 shows the model architecture we build using a baseline model for eg: (Bisk et al., 2016) as the backbone. We also experiment with the state-of-the-art model from Tan and Bansal (2018). Importantly, Bisk et al. (2016) treats both source and target sub-tasks as regression problems while, Tan and Bansal (2018) treats the source sub-task as a classification problem and the target as regression.

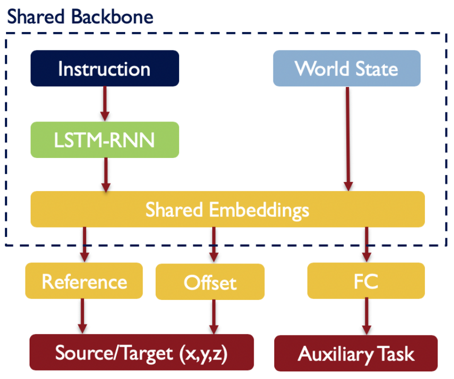

Figure 2: Network architecture with auxiliary tasks. Offset, reference are fully connected (FC) layers. Details of the shared backbone is in (Bisk et al., 2016).

2.1 Data-Augmentation

Most of the instructions involve several spatial relations and a high degree of compositionality. Our data augmentation strategy is designed to (i) teach the model the individual spatial concepts which form components of such compositional instructions. (ii) test if existing models do reasoning by evaluating them on the simpler, generated instructions. We later show this data augmentation strategy benefits both performance on the standard test set and on the generated test-sets. We use multiple surface forms for action words and spatial concepts (eg: *right, east*) based on a small set of common substitutions in the training data [1]. For the challenging unlabeled blocks case, the data augmentation templates cover the following categories (Table 1):
(a) **Fixed-Target (T) vs Fixed-Source (S)**: For T-augmentation, the target location is kept fixed (e.g. *the center of the board*) and for S-augmentation, the source is kept fixed (e.g. *the center block*).
(b) **Absolute (A) vs Relative (R) spatial concepts**: For the A-augmentation, we teach the model to identify concepts like *top right corner*, which depends on the board (fixed) but not on the configuration of the blocks. For R-augmentation, we teach the model to identify concepts like *rightmost*,

[1]The data augmentation templates are described in Appendix A.

which depends on the configuration of other blocks. For the **Labeled** blocks case, we only use one template category that teaches the model relative spatial relations between the named blocks. The generated data is used to pre-train both the models from Bisk et al. (2016) and Tan and Bansal (2018).

Template	Example Sentence
TA	Move the northwest corner block to the center.
TR	Move the leftmost block to the center.
SA	Move the center block to the top left corner.
SR	Move the center block two spaces to the left.
Labeled	Move the BMW block above the Shell block.

Table 1: Templates used for data augmentation. TA, TR, SA, SR corresponds to a combination of the Fixed-Target/-Source with Absolute/Relative augmentation.

2.2 Auxiliary Tasks

We now describe auxiliary tasks which provide explicit signals to the model regarding the components of a complex instruction. For instance, providing feedback regarding the quadrant of the board in which the source block lies in, helps the model learn concepts like *northeast corner*.

Backbone Model: In principle, any model for this task can be used as the backbone with auxiliary branches added. In Fig. 2 we show the model from Bisk et al. (2016) modified for our setting. The hidden state of the LSTM and world state are shared with the auxiliary task branches (described below). The experiments are conducted on the source and target prediction sub-tasks separately. The main branch is trained with mean squared loss. The auxiliary branches use the cross-entropy loss. During the training stage, we alternate among the main prediction branch and the two auxiliary branches.[2] At test time, we only keep the main task branch. For Bisk et al. (2016), evaluation is done in terms of the mean *block-distance*: euclidean distance between the ground truth and model prediction, normalized by the block length, and using accuracy for Tan and Bansal (2018).

Quadrant Auxiliary Task: Aims at teaching the model absolute spatial concepts like *top right corner*. The model is made to predict the answer to *Which quadrant does the source/target belong to?* and is provided feedback as the *top left/ top right/ bottom left/ bottom right quadrant*.

[2]We use the Adam Optimizer (Kingma and Ba, 2014) with a learning rate of 0.001 and 0.0001 for the main and auxiliary branches respectively. We also experimented with multi-task training with a weighted loss on the 3 tasks which gave similar performance gains but takes longer to train.

	Source		Target	
Model	BD	RI%	BD	RI%
B_U	3.47	—	3.70	—
B_U+ Aux.	3.21	7.49	3.44	7.03
B_U+Aux.+ Aug.	**3.11**	10.37	**3.37**	8.92
B_L	0.19	—	1.05	—
B_L+Aux.	0.18	5.26	0.99	5.71
B_L+Aux.+Aug.	**0.17**	10.53	**0.97**	7.62

Table 2: Ablation study of our augmentation approach against the baselines (unlabeled: $\mathbf{B_U}$, labeled: $\mathbf{B_L}$), trained on the full data. **Aux.** denotes auxiliary tasks and **Aug.** denotes data augmentation. **BD** denotes mean *block-distance* (defined in Sec. 2.2) and lower is better. **RI** denotes the relative improvement (in percentage) of each entry over the corresponding baseline performance ($\mathbf{B_U}$,$\mathbf{B_L}$) (Bisk et al., 2016). In the labeled scenario, the gains are small (since $\mathbf{B_L}$ is a strong baseline in this easy task) but are statistically significant.

The hidden state is concatenated with the world state and then passed through a fully-connected layer to solve the four-class classification problem. Training on this auxiliary task jointly with the main task enables the model to learn absolute spatial concepts, such as, *southeast*.

Anchor Auxiliary Task: Aims at teaching the model relative spatial concepts like *leftmost*. The model predicts the answer to *Is block $\#i$ on the top/bottom/left/right of the source (or target)?* with *True or False for each of the four directions top/ bottom/ left/ right*. The hidden state and the world state are passed through a fully-connected layer for an output of size 20×4. For each block that is present on the grid, the model outputs 1 or 0 for each of the 4 directions based on its relative position to the source/target. Training on this auxiliary task jointly with the main task can help the model learn relative spatial concepts. For instance, for the instruction *Move the leftmost block* ..., the model learns that all blocks are to the right of the source block from the received feedback. Both auxiliary tasks are created from the main task by a deterministic function of the world and the target/source location, and requires no extra supervision.

3 Experiments

Here we present empirical evidence that shows the role of (1) pre-training with simpler instructions. (2) joint training with auxiliary tasks. We use the data sets from (Bisk et al., 2016): the un-labeled blocks data set has 2493 training and 720 test examples and the labeled blocks data set has 11871

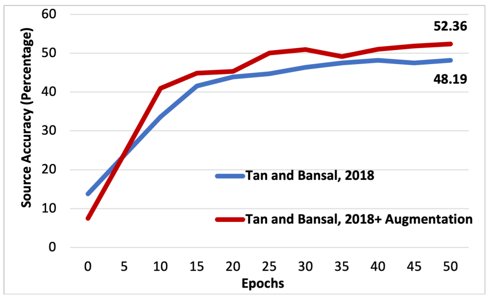

Figure 3: Our proposed augmentation helps the Tan and Bansal (2018) model converge faster to a higher source accuracy on the test set, giving **8.65%** rel. gain.

training and 3177 test examples. We randomly batch the data across the different board configurations, with a batch-size of 9 [3]. The experiments are conducted on the source and target prediction sub-tasks separately. We compare the benefits of data augmentation, task augmentation and a combination of both for each baseline model (B_U and B_L) (Bisk et al., 2016) in Table 2. In Figure 3, we also show the benefits of our proposed augmentation scheme on the SOTA model (Tan and Bansal, 2018), where the source sub-task is evaluated with classification accuracy. We pre-train the model with 2000 generated instructions for both data sets. In the unlabeled case, for source prediction, we use TA and TR templates while, for target prediction we use SA and SR templates (in equal proportion). We also use one-hot encodings for the instruction words following prior work (Bisk et al., 2016).[4]

Figure 4 shows the benefits of our approach for different sizes of training data. The performances are averaged over 5 runs. In both Table 2 and Fig. 4, we observe that we always consistently outperform the baseline for both source and target prediction. In particular, data augmentation and task augmentation bring independent benefits and when combined, yield the best results. The benefit of our approach is more pronounced for less training data.

3.1 Understanding why augmentation helps

Here we show evidence of why data and task augmentation improve the overall performance of the model, focusing on TA and SA augmentation

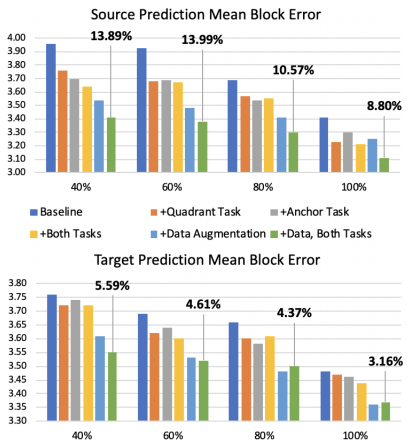

Figure 4: Ablation study of our augmentation approach against the baseline (B_U) for different percentages of training data. B_U is the model trained on the unlabeled blocks data (Bisk et al., 2016) with our batching scheme. The percentage above the green bar shows the relative improvement w.r.t. the dark blue (baseline) bar.

Model	**Source**		**Target**	
	BD	RI%	BD	RI%
B_U	3.12	—	3.59	—
B_U+ Q	2.80	10.26	3.48	3.06
B_U+ Q+ Aug.	**2.76**	11.54	**3.35**	6.69

Table 3: Ablated gains for mean block distance (BD) on the diagnostic subset. B_U: baseline model, Q: quadrant auxiliary task, *Aug* denotes the corresponding data augmentation: TA for Source, SA for Target.

and the quadrant auxiliary task for the un-labeled blocks dataset. We compare the baseline model (B_U), baseline model with the quadrant task branch (B_U+Q) and baseline model with the quadrant task branch and pre-trained with the corresponding augmentation (B_U+Q+Aug.): TA augmentation for Source and SA for Target sub-task. All models are trained on the entire real training data. We first show that auxiliary tasks not only help improve standard test performance (Table 2) but also performance on the generated synthetic instructions. On a diagnostic test set of 1000 TA instructions, the baseline model B_U obtains **3.10** mean B.D. while B_U+Q obtains **2.89**. We further create a diagnostic-subset of the test set by filtering examples that refer to the source/target block using a closed set of quadrant location keywords, e.g.: *northeast*.[5] Table 3 shows the results on this subset. As an example, for identifying the source location in *"The bottom left box moves to the northeast ..."*, (B_U), (B_U+Q) and (B_U+Q+TA) have prediction errors of 6.42, 1.45 and 1.08 respectively. Similarly, for target prediction, (B+Q+SA) has a prediction error

[3]This yields better performance than the configuration-wise batching in Bisk et al. (2016).

[4]**Pretrained Language Embeddings**: We also tried to initialize the LSTM-RNN with pre-trained BERT embeddings and BERT embeddings fine-tuned on the blocks world instructions. We did not observe any improvement on task performances probably because BERT has not been pre-trained on sentences from any similar spatially involved domain.

[5]The full set of keywords is described in Appendix A.

of 2.12 versus the baseline error of 5.20 on *"The block closest to the northwest corner of the table should be near the southwest corner of the table ..."*. On the diagnostic subset, augmentation improves the source accuracy of the Tan and Bansal (2018) model by **5.73%**; Fig. 1 shows an example. These experiments confirm that quadrant task and TA/SA augmentation helps when instructions contain keywords that indicate quadrant information.

4 Conclusion

We showed the benefits of data and task augmentation for instruction understanding on two datasets and two existing models for this task, improving their shortcomings on simple examples.

Acknowledgments

This work was supported by Contracts W911NF-15-1-0461 and FA8750-19-2-0201 with the US Defense Advanced Research Projects Agency (DARPA), the Army Research Office under Grant Number W911NF-20-1-0080, and by ONR Contract N00014-19-1-2620. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Department of Defense or the U.S. Government.

References

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Yonatan Bisk, Kevin J Shih, Yejin Choi, and Daniel Marcu. 2018. Learning interpretable spatial operations in a rich 3d blocks world. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Yonatan Bisk, Deniz Yuret, and Daniel Marcu. 2016. Natural language communication with robots. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 751–761.

Satchuthananthavale RK Branavan, Harr Chen, Luke S Zettlemoyer, and Regina Barzilay. 2009. Reinforcement learning for mapping instructions to actions. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 82–90. Association for Computational Linguistics.

David L Chen and Raymond J Mooney. 2011. Learning to interpret natural language navigation instructions from observations. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*.

Prashant Jayannavar, Anjali Narayan-Chen, and Julia Hockenmaier. 2020. Learning to execute instructions in a minecraft dialogue. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2589–2602.

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

Nikhil Mehta and Dan Goldwasser. 2019. Improving natural language interaction with robots using advice. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1962–1967.

Dipendra Misra, John Langford, and Yoav Artzi. 2017. Mapping instructions and visual observations to actions with reinforcement learning. *arXiv preprint arXiv:1704.08795*.

Anjali Narayan-Chen, Prashant Jayannavar, and Julia Hockenmaier. 2019. Collaborative dialogue in minecraft. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5405–5415.

Hao Tan and Mohit Bansal. 2018. Source-target inference models for spatial instruction understanding. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. 2011. Understanding natural language commands for robotic navigation and mobile manipulation. In *Twenty-fifth AAAI conference on artificial intelligence*.

Kim-Han Thung and Chong-Yaw Wee. 2018. A brief review on multi-task learning. *Multimedia Tools and Applications*, 77(22):29705–29725.

Terry Winograd. 1972. Understanding natural language. *Cognitive psychology*, 3(1):1–191.

A Appendix

A.1 Data Augmentation

- List of actions: *move, place, reposition*
- Mapping of concepts to words:
  1. top-most: *topmost, highest, top-most, top most, uppermost, upper most*
  2. bottom-most: *bottommost, lowest, bottom-most, bottom most, lowermost, lower most*

3. left-most : *far left, farthest left, left most, left-most, leftmost*
4. right-most : *far right, farthest right, right most, right-most, rightmost*
5. center : *center, middle, center of the board, middle of the board*
6. top-right-corner : *northeast corner, northeastern, upper right corner, north east corner, top right corner, upper right corner, upper right hand corner*
7. top-left-corner : *northwest corner, northwestern, upper left corner, north west corner, top left corner, upper left corner, upper left hand corner*
8. bottom-right-corner : *southeast corner, southeastern, lower right corner, south east corner, bottom right corner, lower right corner, lower right hand corner*
9. bottom-left-corner : *southwest corner, southwestern, lower left corner, south west corner, bottom left corner, lower left corner, lower left hand corner*

- For the quantification of spaces we allow $1, 2, 3, 4$.
- For the possible directions we allow : *north/up, south/down, west/left, east/right, northeast, northwest, southeast, southwest*
- For the TA augmentation we use: $[action]$ the $\times$ $random(6, 7, 8, 9)$ $+$ $block$ to the $[center]$
- For TR augmentation we use: $[action]$ the $\times$ $random(1, 2, 3, 4)$ $+$ $block$ to the $[center]$
- For the SA augmentation we use: $[action] \times the$ $center$ $block$ to $the \times random(6, 7, 8, 9)$
- For the SR augmentation we use: $[action]$ $\times$ the $center$ $block$ $\times$ $[1, 2, 3, 4]$ $spaces \times [directions]$
- Here *random* denotes a random choice of the corresponding numbered mappings of concepts to words.
- We place 10 blocks on the board randomly and the remaining 10 block coordinates are set to $(-1, -1, -1)$ in accordance with (Bisk et al., 2016).

A.2 Quadrant Subset Filters

The following keywords are used for filtering the test-set:
Top Left / Upper Left / Northwest / Back Left
Top Right / Upper Right / Northeast
Bottom Left / Lower Left / Southwest / Front Left
Bottom Right / Lower Right / Southeast