Improving Cross-Modal Alignment in Vision Language Navigation via Syntactic Information

Jialu Li **Hao Tan** **Mohit Bansal**
UNC Chapel Hill
{jialuli, airsplay, mbansal}@cs.unc.edu

Abstract

Vision language navigation is the task that requires an agent to navigate through a 3D environment based on natural language instructions. One key challenge in this task is to ground instructions with the current visual information that the agent perceives. Most of the existing work employs soft attention over individual words to locate the instruction required for the next action. However, different words have different functions in a sentence (e.g., modifiers convey attributes, verbs convey actions). Syntax information like dependencies and phrase structures can aid the agent to locate important parts of the instruction. Hence, in this paper, we propose a navigation agent that utilizes syntax information derived from a dependency tree to enhance alignment between the instruction and the current visual scenes. Empirically, our agent outperforms the baseline model that does not use syntax information on the Room-to-Room dataset, especially in the unseen environment. Besides, our agent achieves the new state-of-the-art on Room-Across-Room dataset, which contains instructions in 3 languages (English, Hindi, and Telugu). We also show that our agent is better at aligning instructions with the current visual information via qualitative visualizations.[1]

1 Introduction

Vision-Language Navigation defines the task of requiring an agent to navigate through a visual environment based on natural language instructions (Anderson et al., 2018b; Misra et al., 2018; Chen et al., 2019; Jain et al., 2019; Nguyen and Daumé III, 2019; Thomason et al., 2020). This task poses several challenges. To complete this task, an embodied agent needs to perceive the surrounding environment, understand the given natural language instructions, and most importantly, ground (or align) the instruction in the visual scenes. In this paper, we aim to make one step towards grounding natural language instructions with visual environment via syntax-enriched alignment.

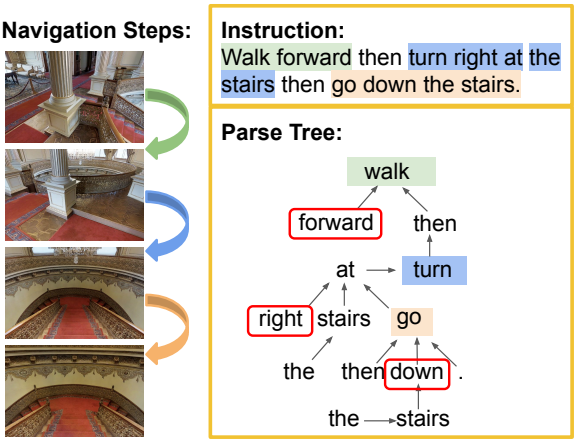

Figure 1: An example in the Room-to-Room task. We generate the dependency parse tree for the instruction. The words are grouped by the head node (highlighted in the tree). Each sub-instruction (i.e., grouped words) corresponds to one step in the navigation with the same color. Modifiers in red boxes can be easily identified from the tree structure.

Recently, several approaches were proposed to solve the Vision-Language Navigation task with better interactions between natural language instructions and visual scenes (Fried et al., 2018; Wang et al., 2019; Landi et al., 2019; Wang et al., 2020a; Huang et al., 2019a; Hu et al., 2019; Majumdar et al., 2020; Ma et al., 2019a; Qi et al., 2020b; Zhu et al., 2020a,c). Some approaches utilize soft attention over individual words for better cross-modal grounding, while others improve co-grounding with better language and vision representation and additional alignment module.

Although these models achieve significant improvement in performance, they do not explicitly consider syntactic linguistic information in their alignment and decision-making. We argue that

[1]Code and models: https://github.com/jialuli-luka/SyntaxVLN

Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1041–1050
June 6–11, 2021.

the syntactic information (e.g., phrases, word functions, modifiers) captured by dependency parse trees is crucial for accurate alignment between the instructions and the environment. As shown in Figure 1, for the instruction *"Walk forward then turn right at the stairs then go down the stairs."*, the dependency parse tree effectively aggregates syntactically-close words together for the agent (e.g., groups phrase information *"Walk forward"* at node *"Walk"*), and each phrase here corresponds to one navigation action. Besides, the dependency tree structure also helps identify modifiers like *"forward"* and *"right"*. This syntactic information helps the agent identify important words, locate phrases (e.g., sub-instructions), and learn a better alignment between the instruction and the visual environment.

Therefore, in this paper, we propose an encoder module that can incorporate simple but important syntactic information from parse trees for the vision-language-navigation task. Our proposed encoder utilizes the ChildSum Tree-LSTM (Tai et al., 2015) over a dependency tree to achieve a syntax-aware representation of the instruction, enabling the agent to focus on more syntactically important words and align not only words but also phrases with the visual scenes.

We conduct experiments on both the Room-to-Room (R2R) dataset (Anderson et al., 2018b) and the Room-across-Room (RxR) dataset (Ku et al., 2020). Empirical results show that our proposed approach significantly improves the performance over the baseline model on success rate and achieves the new state-of-the-art (at the time of submission) on the RxR dataset, which contains instruction in three languages (English, Hindi, and Telugu). Moreover, by using structured information from syntax, we are also able to avoid word-level shallow overfitting of the model and hence achieve better generalization in the unseen environment. Our analysis further shows that our syntax-aware agent has better interpretability and learns better cross-modality matching.

2 Related Work

Visual and Textual Grounding in VLN. In vision-language navigation tasks, visual and textual co-grounding aims to learn the relationship between natural language instructions and the visual environments. A main line of research in VLN utilizes soft attention over individual words for cross-modal grounding in both the natural language instruction and the visual scene (Wang et al., 2018, 2019; Tan et al., 2019; Landi et al., 2019; Xia et al., 2020; Wang et al., 2020b,a; Xiang et al., 2020; Zhu et al., 2020b). Other works improve vision and language representations (Hu et al., 2019; Li et al., 2019; Huang et al., 2019b,a; Hao et al., 2020; Majumdar et al., 2020) and propose an additional progress monitor module (Ma et al., 2019b,a; Ke et al., 2019) and object and action aware modules (Qi et al., 2020b) that aid co-grounding.

The closest work to ours is from Hong et al. (2020), where they use the dependency tree to generate pre-divided sub-instructions, and then propose a shifting module to select and attend to a single sub-instruction at each time-step, which implicitly captures some syntax information. Compared with them, we directly use the dependency tree to explicitly incorporate syntactic information and get syntax-aware instruction representations, hence achieving substantial improvement in generalizing to the unseen environment.

Tree-based Language Representations. Dependency tree provides essential syntactic information for understanding a sentence. Tree-LSTM (Tai et al., 2015) has been widely used to encode parsed tree information and shown improvement over multiple tasks, such as relation extraction (Miwa and Bansal, 2016; Geng et al., 2020), machine translation (Su et al., 2020; Choi et al., 2017; Eriguchi et al., 2016), dialogue (Rao et al., 2019), and language inference (Chen et al., 2017). We are novel in incorporating a dependency tree into the vision-language navigation task via a Tree-LSTM for better phrase-level alignment between the visual environment and language instructions.

3 Method

As illustrated in Figure 2, our base model follows the sequence-to-sequence architecture of previous VLN agents. Our tree-based encoder module is built on top of the strong Environment Drop Agent (Tan et al., 2019). The main difference is that we employ a tree-based language encoder to encode dependency tree information to allow better language grounding. At each time step, we ground all the encoded nodes (i.e., syntax-aware representations) in the dependency tree with the visual information to get the attended textual representation.

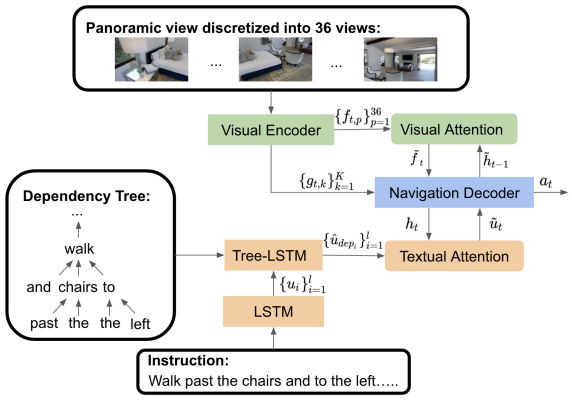

Figure 2: Architecture for our syntax-aware agent.

Generating Dependency Tree Representation with Tree-LSTM. We first generate the dependency parse tree with Stanford CoreNLP (Manning et al., 2014) for English instructions and Stanza Toolkit (Qi et al., 2020a) for Hindi and Telugu instructions. Since the tree-structure is invariant to the children's order (i.e., switching the order of the children of a node doesn't change the tree structure), directly using a Tree-LSTM over an embedding layer may lose important word order information in the instruction. Thus, here we use a bidirectional LSTM with an embedding layer to generate word representations that preserve sequential information of the instruction. Specifically, given an instruction $\{w_i\}_{i=1}^{l}$, we generate syntax-aware representation $\{\hat{u}_{dep_i}\}_{i=1}^{l}$ as:

$$\hat{w}_i = \text{Embedding}(w_i) \quad (1)$$

$$u_1, u_2, ..., u_l = \text{Bi-LSTM}(\hat{w}_1, \hat{w}_2, ..., \hat{w}_l) \quad (2)$$

$$\{\hat{u}_{dep_i}\}_{i=1}^{l} = \text{Tree-LSTM}(\{u_i\}_{i=1}^{l}) \quad (3)$$

Visual Encoder and Navigation Decoder. Given panoramic features $\{f_{t,p}\}_{p=1}^{36}$ and visual representation $\{g_{t,k}\}_{k=1}^{K}$ for K navigable locations at time step t [2], we picks the next viewpoint from K navigable locations as:

$$p_t(a_t = k) = \text{Softmax}_k(g_{t,k}^T W_G \widetilde{h}_t) \quad (4)$$

where $\widetilde{h}_t$ is the context aware hidden states, and W_G is learned weight parameter. Specifically, we compute the $\widetilde{h}_t$ as:

$$\beta_{t,p} = \text{softmax}_p(f_{t,p}^T W_F \widetilde{h}_{t-1}) \quad (5)$$

$$\widetilde{f}_t = \sum_p \beta_{t,p} f_{t,p} \quad (6)$$

$$h_t = \text{LSTM}([\widetilde{f}_t; \widetilde{a}_{t-1}], \widetilde{h}_{t-1}) \quad (7)$$

$$\gamma_{t,i} = \text{softmax}_i(\hat{u}_{dep_i}^T W_U h_t) \quad (8)$$

$$\widetilde{u}_t = \sum_i \gamma_{t,i} \hat{u}_{dep_i} \quad (9)$$

$$\widetilde{h}_t = \tanh(W_M[\widetilde{u}_t; h_t]) \quad (10)$$

where $\widetilde{a}_{t-1}$ is the previous action embedding, $\widetilde{f}_t$ is the attended panoramic representation, and h_t is the decoder hidden state. W_F, W_U, W_M are learned weight parameters. We compute the attended language representation over all dependency node representations which are aware of syntax information.

We use a mixture of imitation learning and reinforcement learning to train the agent. Details can be found in Appendix.

4 Experimental Setup

4.1 Datasets

We evaluate our agent on Room-to-Room (R2R) dataset (Anderson et al., 2018b) and Room-Across-Room (RxR) dataset (Ku et al., 2020). Both datasets are built on Matterport3D simulator (Anderson et al., 2018b). The R2R dataset contains 21567 human-annotated instructions with an average instruction length of 29. The dataset is divided into training set, seen validation set, unseen validation set, and test set. The RxR dataset is an extension of the R2R dataset. The instructions are longer (with an average instruction length of 78), and the instructions are in three languages (i.e., English, Hindi, and Telugu). The RxR dataset follows the same division as the R2R dataset. Details can be found in the Appendix.

4.2 Evaluation Metrics

To evaluate the performance of our model, we use the following evaluation metrics: Success Rate (SR), Success Rate Weighted by Path Length (SPL) (Anderson et al., 2018a), normalized Dynamic Time Warping (nDTW) (Magalhaes et al., 2019), success rate weighted by Dynamic Time Warping (Magalhaes et al., 2019) and Coverage weighted by Length Score (CLS). Detailed description for each metric can be found in Appendix.

[2] Details for generating these features are in Appendix.

Models	Val Seen					Val Unseen				
	SR(%)	SPL	nDTW	sDTW	CLS	SR(%)	SPL	nDTW	sDTW	CLS
EnvDrop	58.3	0.55	0.67	0.52	0.67	45.3	0.42	0.58	0.39	0.58
+LSTM	60.3	0.57	0.69	0.55	0.69	46.4	0.43	0.58	0.40	0.58
+syntax	**62.6**	**0.60**	**0.70**	**0.56**	**0.70**	**49.0**	**0.45**	**0.59**	**0.42**	**0.59**

Table 1: Comparison of the model with and without our tree-based encoder on the seen validation set and unseen validation set of R2R dataset.

Models	Val Seen					Val Unseen				
	SR(%)	SPL	nDTW	sDTW	CLS	SR(%)	SPL	nDTW	sDTW	CLS
EnvDrop (en)	48.1	**0.44**	0.57	0.40	0.61	38.5	0.34	0.51	0.32	0.54
+syntax (en)	**48.1**	0.44	**0.58**	**0.40**	**0.61**	**39.2**	**0.35**	**0.52**	**0.32**	**0.56**
EnvDrop (hi)	49.6	0.45	0.57	0.41	0.61	39.9	0.35	0.49	0.32	0.53
+syntax (hi)	**55.2**	**0.52**	**0.61**	**0.46**	**0.64**	**42.5**	**0.38**	**0.54**	**0.35**	**0.58**
EnvDrop (te)	45.8	0.42	0.56	0.38	0.60	38.3	0.34	0.50	0.31	0.54
+syntax (te)	**49.1**	**0.46**	**0.59**	**0.41**	**0.63**	**38.4**	**0.35**	**0.52**	**0.32**	**0.56**

Table 2: Comparison of the model with the tree-based encoder and without the tree-based encoder on the seen validation set and unseen validation set of RxR dataset.

5 Results and Analysis

5.1 Room-to-Room Dataset

We compare our agent with the baseline agent (Tan et al., 2019)[3] on the R2R test leaderboard. Our syntax-aware agent achieves 47.8% in success rate and 0.45 in SPL, improving the baseline model by 2.1% in success rate and 2% in SPL.

Besides, as shown in Table 1, our syntax-aware agent achieves 3.7% improvement in success rate over the baseline model in validation unseen environment, indicating that explicitly incorporating syntax information can better guide the agent to ground to visual scenes.

We further experiment on R2R dataset to see whether the increase in performance comes from more model parameters. Compared with the model that uses a 2-layer LSTM (+LSTM), we can see that our syntax-aware model still achieves 2.6% increase in success rate in validation unseen environment. This result validates the effectiveness of incorporating syntax information.

5.2 Room-Across-Room Dataset

We first compare our agent with the baseline agent in RxR paper (Ku et al., 2020) on the RxR test leaderboard. Our syntax-aware agent ("SAA (mono)" on the leaderboard) outperforms the baseline in all metrics, improving the nDTW score by 5.73% and success rate by 9.98%. Moreover, We compare our agent with the baseline on RxR validation set. As shown in Table 2, in all three languages, our syntax-aware agent achieves higher success rate than the baseline agent in all metrics on validation unseen set. Specifically, our model gets 2.6% improvement in Hindi instructions in terms of success rate. For English and Telugu instructions, our model gets smaller improvement – 0.7% and 0.1% respectively. One reason for these results could be the correlation with the quality of the dependency parser in that language. Besides, compared with the baseline in Ku et al. (2020), our agent achieves the new state-of-the-art on RxR dataset (at the time of submission).

5.3 Qualitative Analysis

As shown in Fig 3 and Fig 4, we illustrate a qualitative example to show that our agent learns better alignment between instruction and visual scenes. As shown in Fig 4, given instruction *"Walk past the shelves and out of the garage. Stop in front of the opening to the wine cellar."*, the baseline agent tends to focus on the same phrases (e.g., "past the shelves", "wine cellar") during navigation. This suggests that the baseline agent is not able to learn a good alignment between the instruction and the current visual scene. However, as shown in Figure 3, our dependency-based agent can successfully identify the correct parts of the instruction that are correlated with the current visual scenes, and picks the next action with fidelity. At the beginning of navigation, our agent focuses on the first sub-instruction "walk past the shelves". Then "out of the garage" gradually becomes the most important phrase, indicating that the agent should go out of the garage after passing the shelves. When

[3]The baseline is the re-implementation of their model without back translation based on code: https://github.com/airsplay/R2R-EnvDrop

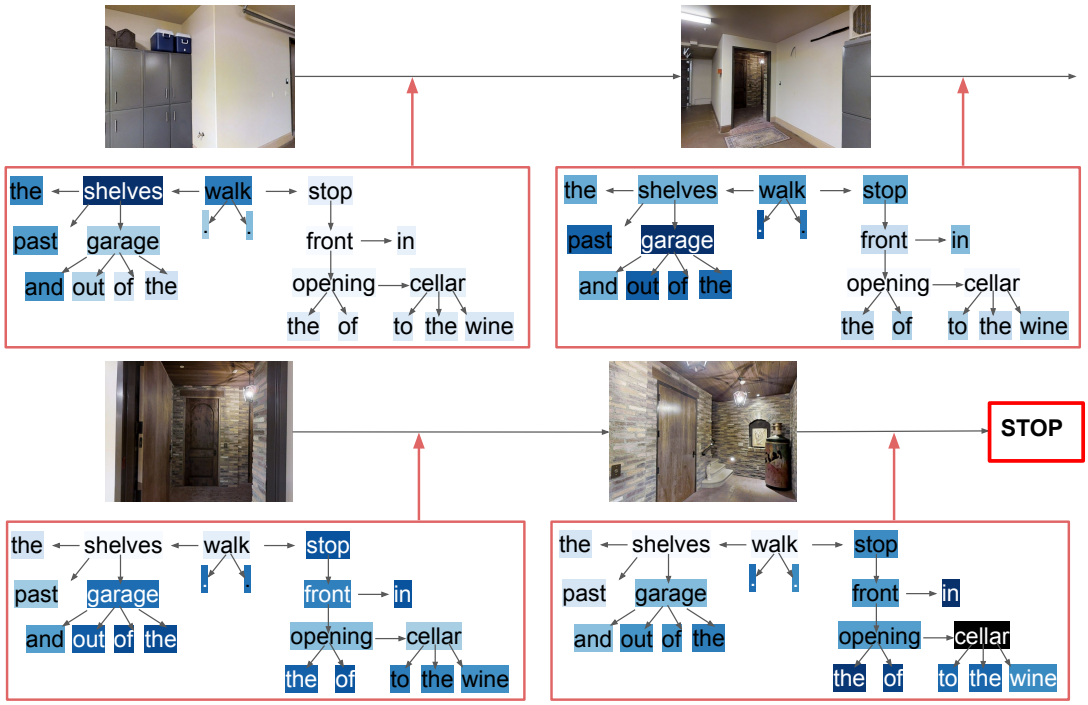

Figure 3: The weights for the grounded instructions for our syntax-aware model.

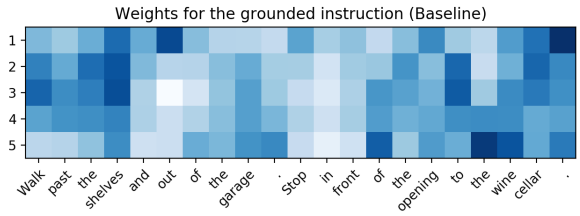

Figure 4: The weights for the grounded instructions for baseline model.

the agent sees the wine sculpture, it infers that the opening is for wine cellar and stops in front of the opening near the wine sculpture.

5.4 Implementation Variants

Since the goal of our paper is to explore the role of syntax information in vision-language navigation, we try several implementation variants to include syntax information. First, we try to use mean-pooling instead of Tree-LSTM to encode the dependency tree structure. This implementation variant decreases the performance by around 3% in terms of success rate on validation unseen environments. Besides, we explore whether syntax information from a constituency tree can also help with the instruction following and navigation. Similar to how we incorporate dependency tree information, we use a Tree-LSTM to encode the constituency tree information. However, the performance decreases around 2% in terms of success rate in validation unseen environments, indicating that syntax information extracted from dependency tree is more beneficial for the vision-and-language navigation task.

6 Conclusion

In this paper, we presented a tree-based encoder module that incorporates phrase-level information from parse trees. We demonstrated that syntax information can help the agent learn a better alignment between instruction and visual scenes, and generalize better to unseen environments. Our experiments on Room-to-Room dataset and Room-Across-Room dataset both suggest that incorporating syntax information encoded by our tree-based encoder module can significantly improve the performance over baseline VLN models.

7 Ethical Considerations

Vision-Language Navigation is the task that requires an agent to navigate through a 3D environment based on given natural language instructions. An agent that can interact with the environment based on instructions can be used in many real-world applications, for example, a home service robot can bring things to the owner based on instruction, making people's life easier. However,

when deployed in the real world, even if the agent can navigate successfully based on the instruction, it might still need further human assistance to keep working successfully (e.g., a home-cleaning robot might be stuck in the corner of the room and cannot get out by itself).

The performance on the validation set with the unseen environment is much lower than the seen environment. Change in environment will significantly influence the performance of the agent. When the agent is deployed in an unseen environment, it will have a higher probability of failure, wasting energy and time. A further pre-exploration of the environment will be needed for better performance of the agent when deployed to real-world applications. Moreover, our agent relies on the quality of the dependency parser to some extent. Though we achieve improvement in all three languages when using the dependency tree information, the agent in Hindi and English benefit most from the syntax information because of the best available parser for these languages.

8 Acknowledgement

We thank the reviewers for their helpful discussions. This work was supported by ARO-YIP Award W911NF-18-1-0336, DARPA MCS Grant N66001-19-2-4031, a Google Focused Research Award, and a Bloomberg Data Science Ph.D. Fellowship. The views, opinions, and/or findings contained in this article are those of the authors and not of the funding agency.

References

Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. 2018a. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. 2018b. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3674–3683.

Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. 2019. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12538–12547.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668.

Jihun Choi, Kang Min Yoo, and Sang-goo Lee. 2017. Learning to compose task-specific tree structures. *arXiv preprint arXiv:1707.02786*.

Akiko Eriguchi, Kazuma Hashimoto, and Yoshimasa Tsuruoka. 2016. Tree-to-sequence attentional neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 823–833.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. 2018. Speaker-follower models for vision-and-language navigation. In *Advances in Neural Information Processing Systems*, pages 3314–3325.

ZhiQiang Geng, GuoFei Chen, YongMing Han, Gang Lu, and Fang Li. 2020. Semantic relation extraction using sequential and tree-structured lstm with attention. *Information Sciences*, 509:183–192.

Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. 2020. Towards learning a generic agent for vision-and-language navigation via pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13137–13146.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. 2012. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8).

Yicong Hong, Cristian Rodriguez, Qi Wu, and Stephen Gould. 2020. Sub-instruction aware vision-and-language navigation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3360–3376.

Ronghang Hu, Daniel Fried, Anna Rohrbach, Dan Klein, Trevor Darrell, and Kate Saenko. 2019. Are you looking? grounding to multiple modalities in vision-and-language navigation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6551–6557.

Haoshuo Huang, Vihan Jain, Harsh Mehta, Jason Baldridge, and E. Ie. 2019a. Multi-modal discriminative model for vision-and-language navigation. *ArXiv*, abs/1905.13358.

Haoshuo Huang, Vihan Jain, Harsh Mehta, Alexander Ku, Gabriel Magalhaes, Jason Baldridge, and Eugene Ie. 2019b. Transferable representation learning in vision-and-language navigation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7404–7413.

Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. 2019. Stay on the path: Instruction fidelity in vision-and-language navigation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1862–1872.

Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha Srinivasa. 2019. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6741–6749.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. 2020. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. *arXiv preprint arXiv:2010.07954*.

Federico Landi, Lorenzo Baraldi, Marcella Cornia, Massimiliano Corsini, and Rita Cucchiara. 2019. Perceive, transform, and act: Multi-modal attention networks for vision-and-language navigation. *arXiv preprint arXiv:1911.12377*.

Xiujun Li, C. Li, Qiaolin Xia, Yonatan Bisk, A. Çelikyilmaz, Jianfeng Gao, Noah A. Smith, and Yejin Choi. 2019. Robust navigation with language pretraining and stochastic sampling. In *EMNLP/IJCNLP*.

Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. 2019a. Self-monitoring navigation agent via auxiliary progress estimation. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. 2019b. The regretful agent: Heuristic-aided navigation through progress estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6732–6740.

Gabriel Magalhaes, Vihan Jain, Alexander Ku, Eugene Ie, and Jason Baldridge. 2019. Effective and general evaluation for instruction conditioned navigation using dynamic time warping. *arXiv preprint arXiv:1907.05446*.

Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. 2020. Improving vision-and-language navigation with image-text pairs from the web. *arXiv preprint arXiv:2004.14973*.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Dipendra Kumar Misra, Andrew Bennett, Valts Blukis, Eyvind Niklasson, Max Shatkhin, and Yoav Artzi. 2018. Mapping instructions to actions in 3d environments with visual goal prediction. In *EMNLP*.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937.

Khanh Nguyen and Hal Daumé III. 2019. Help, anna! visual navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning. 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)*, pages 684–695.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020a. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Yuankai Qi, Zizheng Pan, Shengping Zhang, Anton van den Hengel, and Qi Wu. 2020b. Object-and-action aware model for visual language navigation. *arXiv preprint arXiv:2007.14626*.

Jinfeng Rao, Kartikeya Upasani, Anusha Balakrishnan, Michael White, Anuj Kumar, and Rajen Subba. 2019. A tree-to-sequence model for neural nlg in task-oriented dialog. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 95–100.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge.

International Journal of Computer Vision (IJCV), 115(3):211–252.

Chao Su, Heyan Huang, Shumin Shi, Ping Jian, and Xuewen Shi. 2020. Neural machine translation with gumbel tree-lstm based encoder. *Journal of Visual Communication and Image Representation*, page 102811.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 1556–1566.

Hao Tan, Licheng Yu, and Mohit Bansal. 2019. Learning to navigate unseen environments: Back translation with environmental dropout. 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 2610–2621.

Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. 2020. Vision-and-dialog navigation. In *Conference on Robot Learning*, pages 394–406.

Hanqing Wang, Wenguan Wang, Tianmin Shu, Wei Liang, and Jianbing Shen. 2020a. Active visual information gathering for vision-language navigation. *arXiv preprint arXiv:2007.08037*.

Hu Wang, Qi Wu, and Chunhua Shen. 2020b. Soft expert reward learning for vision-and-language navigation. *arXiv preprint arXiv:2007.10835*.

Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. 2019. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6629–6638.

Xin Wang, Wenhan Xiong, Hongmin Wang, and William Yang Wang. 2018. Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 37–53.

Qiaolin Xia, Xiujun Li, Chunyuan Li, Yonatan Bisk, Zhifang Sui, Jianfeng Gao, Yejin Choi, and Noah A Smith. 2020. Multi-view learning for vision-and-language navigation. *arXiv preprint arXiv:2003.00857*.

Jiannan Xiang, Xin Eric Wang, and William Yang Wang. 2020. Learning to stop: A simple yet effective approach to urban vision-language navigation. *arXiv preprint arXiv:2009.13112*.

Fengda Zhu, Yi Zhu, Xiaojun Chang, and Xiaodan Liang. 2020a. Vision-language navigation with self-supervised auxiliary reasoning tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10012–10022.

Wang Zhu, Hexiang Hu, Jiacheng Chen, Zhiwei Deng, Vihan Jain, Eugene Ie, and Fei Sha. 2020b. Babywalk: Going farther in vision-and-language navigation by taking baby steps. In *ACL*, pages 2539–2556.

Yi Zhu, Fengda Zhu, Zhaohuan Zhan, Bingqian Lin, Jianbin Jiao, Xiaojun Chang, and Xiaodan Liang. 2020c. Vision-dialog navigation by exploring cross-modal memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

A Appendix

A.1 Problem Setup

Vision-Language navigation requires an agent to navigate through a 3D environment to a target location based on a given natural language instruction. Formally, the natural language instruction is a sequence of words $\{w_i\}_{i=1}^{l}$, where l is the length of the instruction and w_i is ith word in the sequence. At each time step, the agent perceives a panoramic view of the current viewpoint. Besides, the agent has access to a set of navigable locations $\{l_{t,k}\}_{k=1}^{K}$, where K is the total number of reachable locations from the current viewpoint. The agent needs to select an action a_t from the list of navigable viewpoints $\{l_{t,k}\}_{k=1}^{K}$ based on the given instruction, navigation history and current panoramic views. If the viewpoint selected from the list is the same as the current viewpoint, the agent predicts a "STOP" action.

A.2 Model Details

Visual Encoder. Same as previous work in VLN, at time step t, we discretize the panoramic view into 36 single views $\{o_{t,p}\}_{p=1}^{36}$. Each single view is a RGB image, annotated with its angles of heading and elevation $(\theta_{t,p}, \phi_{t,p})$. Each RGB image is encoded with a pre-trained ResNet-152 (He et al., 2016) on ImageNet (Russakovsky et al., 2015). A four dimension orientation feature $(cos\theta_{t,p}, sin\theta_{t,p}, cos\phi_{t,p}, sin\phi_{t,p})$ is concatenated with the ResNet feature to form the final representation for the view p of a panoramic $\{f_{t,p}\}_{p=1}^{36}$. Similarly, we get the visual representation $\{g_{t,k}\}_{k=1}^{K}$ for K navigable locations $\{l_{t,k}\}_{k=1}^{K}$ at time step t.

A.3 Training

We use a mixture of imitation learning and reinforcement learning to train the agent.

Imitation Learning. During training, instead of navigating to the predicted action at each time step, teacher-forcing is used to determine which navigable viewpoint to pick. Given the shortest path between the start point and target point, at each time step t, the agent tries to imitate the teacher action $a_t^\star$ by minimizing the negative log probability:

$$L_{IL} = \sum_t -a_t^\star log p_t \tag{11}$$

Reinforcement Learning. We combine imitation learning with reinforcement learning to learn a more generalizable agent. Since the teacher path is the shortest path between the start point and the target point, there is no guarantee that the teacher path is the same as indicated by the given instruction. Thus, reinforcement learning is applied for better instruction following and state exploration. At each time step t, the agent samples an action a_t from the predicted distribution $p_t(a_t)$. At each time step, if the agent moves closer to the target viewpoint, a positive reward +1 is given, otherwise the agent receives a negative reward -1. When the agent predicts the "STOP" action, the agent will receive a +3/-3 reward based on whether the agent is within 3m from the target viewpoint. We use Actor-Critic (Mnih et al., 2016) to train the agent. The loss of reinforcement learning is:

$$L_{RL} = \sum_t (R_t - R_{b_t}) log p_t(a_t) + \eta H(p_t(a_t)) \tag{12}$$

where R_t is the discounted future cumulative rewards at time step t, R_{b_t} is the expected cumulative rewards (baseline) approximated by the value function V, $H(p_t(a_t))$ is the entropy term for regularization. Specifically,

$$R_t = r_t + \sum_{i=1}^{T-t} \gamma^i r_{t+i} \tag{13}$$

$$R_{b_t} = V(h_t) = W_{v_2} \sigma(W_{v_1} h_t) \tag{14}$$

where σ is the ReLU activation function, and r_t is the immediate reward we defined earlier. The value function is trained with L2 loss:

$$L_V = \frac{1}{2}(R_t - R_{b_t})^2 \tag{15}$$

We optimize a mixture loss of imitation learning and reinforcement learning:

$$L_{MIX} = (L_{RL} + L_V) + \lambda L_{IL} \tag{16}$$

A.4 Dataset

R2R Dataset. The R2R dataset contains 21567 human annotated instructions with an average instruction lengths of 29. The training set contains 14025 instructions in 61 environments. The seen validation set contains 1020 instructions in the same 61 environments as the training set. The unseen validation set contains 2349 instructions in 11 environment which is not included in the training set. The test set contains 4173 instructions in 18 environments.

RxR Dataset. The RxR dataset is an extension to the R2R dataset, where the instructions are longer and in languages other than English (Hindi and Telugu). It contains 126069 instructions with an average instruction length of 78. Besides, different from R2R that only contains guide path (i.e., the shortest path between the start point and the target point), RxR pairs each guide path with a human-annotated follower path (i.e., the path that human generates following the instruction). We only use the guide path to train the agent in this paper. The seen and unseen environment split is the same as in the R2R dataset. There are 16522 paths in total, and each path is annotated in 3 languages. The training set contains 11089 paths, the seen validation set contains 1232 paths, the unseen validation contains 1517 paths, and the test set contains 2684 paths.

A.5 Evaluation Metrics

To evaluate the performance of our model, we use the following evaluation metrics: (1) Success Rate (SR): If the agent stops less than 3m from the target location, we consider the navigation as a success. (2) Success Rate Weighted by Path Length (SPL) (Anderson et al., 2018a): This metric penalizes long trajectories (e.g., find the target using beam search over the environment graph). (3) normalized Dynamic Time Warping (nDTW) (Magalhaes et al., 2019): This metric penalizes deviations from the reference path. (4) success rate weighted by Dynamic Time Warping (sDTW) (Magalhaes et al., 2019): This metric constraints nDTW only to successful navigation and considers path fidelity and agent success. (5) Coverage weighted by Length

Score (CLS) (Jain et al., 2019): Similar to nDTW, this metric also encourages path fidelity.

A.6 Implementation Details

We generate the dependency parse tree for English using Stanford CoreNLP (Manning et al., 2014). We use Stanza Toolkit (Qi et al., 2020a) to generate the dependency parse tree for Hindi and Telugu. For both baseline model and our syntax-aware model, the learned word embedding size is 256 and the dimension of the action embedding is 128. We set the hidden size for the bi-directional LSTM to be 256. For syntax-aware model, the hidden size for the bi-LSTM and Tree-LSTM in the language encoder is 512. We use ResNet-152 (He et al., 2016) to extract the 2048 dimensional image feature. We use RMSProp (Hinton et al., 2012) as the optimizer with learning rate 1e-4 and batch size 64. The weight λ we use to combine imitation learning loss and reinforcement learning loss is set to be 0.2. In reinforcement learning, the discount factor γ is 0.9 and the entropy weight η is 0.01. During training, we set the max action length to be 35 for R2R dataset and 70 for RxR dataset. We train both agents on R2R for 80,000 iterations. We train both agents on RxR for 200,000 iterations. The baseline model contains approximate 6 million parameters. Our syntax-aware model contain approximate 8 million parameters.