Not All Actions Are Equal: Learning to Stop in Language-Grounded Urban Navigation

Jiannan Xiang[*]
USTC
szxjn@mail.ustc.edu.cn

Xin Wang(✉)
UC Santa Barbara
xwang@cs.ucsb.edu

William Yang Wang
UC Santa Barbara
william@cs.ucsb.edu

Abstract

Vision-and-Language Navigation (VLN) is a natural language grounding task where an agent learns to follow natural language instructions to navigate to specified destinations in photo-realistic environments. A successful agent should not only produce high-fidelity navigation trajectories matched with instructions but also stop at the correct location. However, existing methods mainly focus on alignment between trajectories and instructions, but treat STOP action equally as other actions in the action space. This results in undesirable behaviors that the agent often fails to stop at designated places even though it might be on the right path. Therefore, we propose a two-branch policy model to treat STOP and the other actions differently and thus learn to stop better. Particularly, a direction decider is used to choose directions at key points and a stop indicator is employed to produce stop or non-stop signals. Moreover, to alleviate the unbalanced occurrence of STOP and other actions, we use a weighted cross-entropy loss to force the agent to pay more attention to STOP. We demonstrate that our two-branch policy model can not only make better decisions on where to go next but also stop more accurately. Experiments show that our approach achieves new state-of-the-art results on the TOUCHDOWN dataset, outperforming the baseline model by 5.73% (absolute improvement) on Success weighted by Edit Distance (SED).

1 Introduction

Recently, Vision-and-Language Navigation [1, 2], where an agent is required to navigate in real photo-realistic environments by following natural language instructions, is receiving more and more attention. This task is particularly challenging, as the agent must have a deep understanding of both surrounding visual scene and natural language instructions and aligning them well to maximize the supervision from the instructions. Various approaches have been proposed for VLN [14, 12, 6, 10, 9, 3, 15], and most of them focus on the alignment between trajectories and instructions and learn a policy that treats all actions equally. However, this can cause an undesirable behavior that the agent fails to stop at the target although it might be on the right path, because the STOP action is severely underestimated.

We argue that STOP is more important than other actions and deserves special treatment. First, contrary to errors on other actions, which can be fixed later in the journey, the price of stopping at a wrong location is high because producing STOP terminates the episode and there will be no chance to fix a wrong stop. Second, The statistical count of STOP is much lower than other actions as it only appears once per episode. Thus STOP will receive less attention if we treat all actions equally and ignore the difference of occurrence frequency. Moreover, STOP and other actions need different understandings of the dynamics between the instruction and the visual scene. Both require the alignment between trajectories and instructions, but STOP would emphasize the completeness

[*] Work was done when the first author was interning at UC Santa Barbara.

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

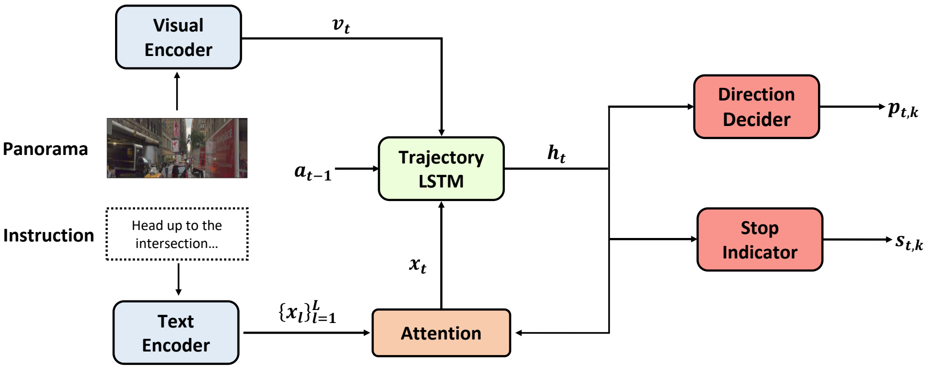

Figure 1: Overview of our two-branch policy model.

of the instruction and the matching between the inferred target and the surrounding scene, while choosing directions requires a planning ability to imagine the future trajectory.

Therefore, in this paper, we introduce the *learning to stop* principle to address the problems above. Specifically, we propose a two-branch policy model, consisting of a *Stop Indicator* to determine whether to stop and a *Direction Decider* to choose directions to go at each time step. Furthermore, we weigh STOP more than other actions with a Weighted Cross-Entropy Loss function for the two-branch policy, forcing the agent to pay more attention to where to stop. We conduct the experiments on a challenging language-grounded street-view navigation dataset, TOUCHDOWN [2]. Experimental results show that our proposed approach significantly improves the performance over the baseline model on all metrics and achieves the new state-of-the-art on the TOUCHDOWN dataset.

2 Approach

Fig. 1 illustrates the framework of our two-branch policy model, which is adapted from RCONCAT [2]. We detail each component below.

2.1 Visual and Text Encoder

As shown in the left part of the Fig. 1, we use two encoders for encoding visual scene and language instruction respectively. Specifically, for visual part, we apply a convolutional neural network (CNN) [7] as the visual encoder to extract visual representation $\boldsymbol{v}_t$ from current visual scene at time step t. For the text part, we adopt an LSTM [4] as text encoder to get the instruction representation $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_l\}$, where $\boldsymbol{x}_l$ denotes the l-th word feature vectors in the instruction. We then use a soft-attention [13] module to get the grounded textual feature $\boldsymbol{x}_t$ at time step t:

$$\alpha_{t,l} = softmax((\boldsymbol{W}_x \boldsymbol{h}_{t-1})^{\boldsymbol{T}} \boldsymbol{x}_l) \quad and \quad \boldsymbol{x}_t = \sum_l \alpha_{t,l} \boldsymbol{x}_l \tag{1}$$

where $\boldsymbol{W}_x$ denotes parameters to be learnt, $\alpha_{t,l}$ denotes attention weight over l-th feature vector at time t, and h_{t-1} denotes hidden context of previous time step. Based on visual representation $\boldsymbol{v}_t$, grounded textual feature $\boldsymbol{x}_t$ at time step t, and previous action embedding $\boldsymbol{a}_{t-1}$, the agent produces the hidden context of the current step $\boldsymbol{h}_t$:

$$\boldsymbol{h}_t = LSTM([\boldsymbol{x}_t, \boldsymbol{v}_t, \boldsymbol{a}_{t-1}]) \tag{2}$$

2.2 Two-Branch Policy

Unlike the existing methods that view all the actions equivalently important, we propose a two-branch policy model to learn where to stop and which directions to go next with separate modules.

Direction Decider The direction decider is employed to select actions from the action space (*go forward, turn left, and turn right*). Empirically, we observe that when navigating in urban

Table 1: Experimental results on development and test sets.

	Development					Test				
Method	TC↑	SPD↓	SED↑	CLS↑	SDTW↑	TC↑	SPD↓	SED↑	CLS↑	SDTW↑
Random	0.15	26.63	0.05	4.65	0.06	0.36	26.94	0.01	4.44	0.00
RConcat [2]	10.49	19.99	10.10	47.44	9.96	12.07	19.07	11.75	48.89	11.67
Ours										
ARC	15.33	18.61	14.62	48.56	14.48	14.13	19.41	13.62	48.02	13.50
ARC + two-branch	**19.35**	**16.87**	**18.77**	**55.99**	**18.72**	**17.83**	**18.32**	**17.48**	**54.55**	**17.47**

environments, the agent only needs to choose directions at the intersections it encounters in the journey. Here we define nodes with more than two neighbors as intersections. Therefore, we view these intersections as key points on the road and assume that the direction decider only needs to choose directions at key points and always goes forward otherwise. So at time step t, if the agent is at a key point, it will be activated and takes the hidden context $\boldsymbol{h}_t$ as well as a learned time embedding $\boldsymbol{t}$ as input and outputs the probability of each action in its action space:

$$p_{t,k} = softmax(g_1([\boldsymbol{h}_t, \boldsymbol{t}])) \tag{3}$$

where g_1 is a linear layer and $\hat{p}_{t,k}$ is the probability of each action at time step t.

Stop Indicator The stop indicator produces stop or non-stop signals at every time step. At time step t, the stop indicator takes hidden context $\boldsymbol{h}_t$ and time embedding $\boldsymbol{t}$ as input and outputs the probabilities of stop and non-stop signals:

$$s_{t,1}, s_{t,2} = softmax(g_2([\boldsymbol{h}_t, \boldsymbol{t}])) \tag{4}$$

where g_2 is a linear layer, and $s_{t,1}$ as well as $s_{t,2}$ are the probabilities of non-stop and stop signals at time step t, respectively. If the stop indicator produces stop signal, the agent will stop immediately. Otherwise, the direction decider will choose a direction to go next.

2.3 Learning

In the Touchdown [2] dataset, every example is a pair of a ground-truth trajectory from starting point z_{start} to target point z_{target} in the urban environment and a natural language instruction describing the trajectory. Unlike R2R [1] dataset, the trajectories in Touchdown are not the shortest paths from z_{start} to z_{target}, but have more twists and turns. Therefore, we use Teacher-Forcing [8] method to train the model. We have two loss functions, $\mathcal{L}_{direction}$ and $\mathcal{L}_{stop}$, for direction decider and stop indicator, respectively. $\mathcal{L}_{direction}$ is a regular cross-entropy loss function

For the stop indicator, we use a weighted cross-entropy loss, where we assign a greater weight for the stop signal in the loss function and therefore force the agent to pay more attention to the stop action (which appears much less frequently than non-stop), in formula,

$$\mathcal{L}_{stop} = \sum_t -o_{t,1} log(s_{t,1}) - \lambda o_{t,2} log(s_{t,2}) \tag{5}$$

where $o_{t,1}$ and $o_{t,2} = 1 - o_{t,1}$ are the ground-truth non-stop and stop signals, and λ is the weight for the stop signal $o_{t,2}$. Finally, the agent is optimized with a weighted sum of two loss functions:

$$\mathcal{L}_{loss} = \gamma \mathcal{L}_{direction} + (1 - \gamma) \mathcal{L}_{stop} \tag{6}$$

where γ is the weight balancing the two losses.

3 Experiments

3.1 Experimental Settings

Touchdown Dataset We evaluate our approach on the Touchdown dataset [2] for vision-and-language navigation in real-world urban environment. The navigation environment in the dataset includes 29,641 panoramas and 61,319 edges from New York City, where panoramas are connected in a graph-like structure with undirected edges. The dataset contains 9,326 examples of navigation tasks, which are pairs of ground-truth trajectory and natural language instruction describing the trajectory. As is reported in [2], the dataset is split into training (6,526 examples), development (1,391) and test (1,409) sets.

Table 2: Ablation study results on the development set.

		Development				
#	Model	TC↑	SPD↓	SED↑	CLS↑	SDTW↑
1	ARC + two-branch	**19.35**	**16.87**	**18.77**	**55.99**	**18.72**
2	- one branch	15.40	18.33	14.92	52.00	14.86
3	- no key points	15.18	18.17	14.55	51.67	14.44
4	- no weighting	12.65	21.60	12.22	47.91	12.20

Evaluation Metrics Following [2], We report three evaluation metrics for the VLN task in urban environments: Task Completion (TC), Shortest-path Distance (SPD) and Success weighted by Edit Distance (SED). We also add another two metrics evaluating the alignment between trajectories produced by agent and the natural language instructions: Coverage weighted by Length Score (CLS) [5] and Success weighted by normalized Dynamic Time Warping (SDTW) [11].

3.2 Experimental Results

Table 1 shows the results comparison between our approach and the baselines: (1) Random: randomly take actions at each time step. (2) RCONCAT the best-performing baseline model as reported in the original dataset paper [2]. We first modify the RCONCAT model by equipping it with an attention mechanism on instruction representation to get our Attention-RConcat (ARC) model that outperforms RCONCAT. Then we integrate ARC with our proposed two-branch policy module, which further boosts the performances on all metrics and achieves the best results on both development and test sets. The consistent improvements in goal-oriented metrics (TC and SED) and path alignment metrics (CLS and SDTW) validate that our two-branch policy model learns not only where to stop but also where to go better.

3.3 Ablation Study

We conduct an ablation study to illustrate the effect of each component on the development set in Table 2. Row 2-4 shows the influence of each component by removing them respectively from the final model (ARC with two-branch policy module). Evidently, removing any of the components results in worse performance, which proves the indispensability of all components in our model.

Row 2 shows the results of ARC with only one policy module which will disable *turn left* and *turn right* actions when the agent is not at key points. The results evaluate the effectiveness of the two-branch structure for providing different sub-policies for STOP and other actions. Row 3 shows the results of the model whose Direction Decider makes decisions at every time step instead of only at key points. The results are intuitive because when navigating in urban environments, allowing the agent to choose directions at non-key points, where the agent mostly just goes forward, will produce interference to the agent and thus decreases its ability to choose right directions at key points. Row 4 shows the results where the weight of the stop signal is the same as the non-stop signal in the loss function of Stop Indicator. The worst results validate the importance of STOP action. When stop and non-stop signals are treated equally, the agent will pay more attention to non-stop because of its higher occurrence frequency than stop.

4 Conclusion

In this paper, we investigate the importance of the STOP action and study how to learn a policy that can not only make better decisions on where to go but also stop more accurately. We propose a two-branch policy module for the vision-and-language navigation task that is situated in street-view urban environments and validate its effectiveness. We believe that our two-branch policy module is modular and generalizable to be plugged into other VLN models and further boost their ability to learn to stop.

References

[1] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. 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, 2018.

[2] Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. 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, 2019.

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

[4] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[5] Vihan Jain, Gabriel Magalhaes, Alex Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. Stay on the path: Instruction fidelity in vision-and-language navigation. *arXiv preprint arXiv:1905.12255*, 2019.

[6] Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha Srinivasa. 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, 2019.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[8] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[9] Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. Self-monitoring navigation agent via auxiliary progress estimation. *arXiv preprint arXiv:1901.03035*, 2019.

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

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

[12] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. *arXiv preprint arXiv:1904.04195*, 2019.

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[14] Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. 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, 2019.

[15] Xin Wang, Wenhan Xiong, Hongmin Wang, and William Yang Wang. 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, 2018.