VLN-PETL: Parameter-Efficient Transfer Learning for Vision-and-Language Navigation

Yanyuan Qiao Zheng Yu Qi Wu[*]
Australian Institute for Machine Learning, The University of Adelaide
{yanyuan.qiao,zheng.yu,qi.wu01}@adelaide.edu.au

Abstract

The performance of the Vision-and-Language Navigation (VLN) tasks has witnessed rapid progress recently thanks to the use of large pre-trained vision-and-language models. However, full fine-tuning the pre-trained model for every downstream VLN task is becoming costly due to the considerable model size. Recent research hotspot of Parameter-Efficient Transfer Learning (PETL) shows great potential in efficiently tuning large pre-trained models for the common CV and NLP tasks, which exploits the most of the representation knowledge implied in the pre-trained model while only tunes a minimal set of parameters. However, simply utilizing existing PETL methods for the more challenging VLN tasks may bring non-trivial degeneration to the performance. Therefore, we present the first study to explore PETL methods for VLN tasks and propose a VLN-specific PETL method named VLN-PETL. Specifically, we design two PETL modules: Historical Interaction Booster (HIB) and Cross-modal Interaction Booster (CIB). Then we combine these two modules with several existing PETL methods as the integrated VLN-PETL. Extensive experimental results on four mainstream VLN tasks (R2R, REVERIE, NDH, RxR) demonstrate the effectiveness of our proposed VLN-PETL, where VLN-PETL achieves comparable or even better performance to full fine-tuning and outperforms other PETL methods with promising margins. The source code is available at https://github.com/YanyuanQiao/VLN-PETL

1. Introduction

Large-scale pre-trained models have shown remarkable success in both computer vision (CV) and natural language processing (NLP) domains, and have largely improved the performance of a variety of visio-linguistic tasks [17, 32, 44]. These models follow a standard pretrain-and-finetune paradigm, which first pretrains the model on large-scale un-

[*]Corresponding author

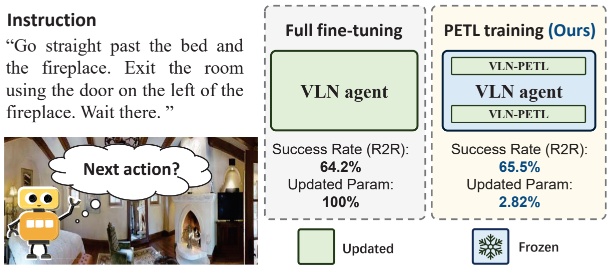

Instruction
"Go straight past the bed and the fireplace. Exit the room using the door on the left of the fireplace. Wait there. "
Next action?
Full fine-tuning
VLN agent
Success Rate (R2R): 64.2%
Updated Param: 100%
PETL training (Ours)
VLN-PETL
VLN agent
VLN-PETL
Success Rate (R2R): 65.5%
Updated Param: 2.82%
Updated
Frozen

Figure 1: Comparison of full fine-tuning and our proposed PETL training for VLN tasks. By updating only a small subset of parameters, our proposed VLN-PETL can achieve a comparative performance compared to full fine-tuning

labeled data and then finetunes it on each downstream task. Since the size of such models is growing rapidly [4, 39], even fully finetuning and storing a copy of the entire pre-trained model for each downstream becomes costly.

To alleviate this problem, Parameter-Efficient Transfer Learning (PETL) has been proposed as an alternative training strategy [3, 10, 14, 15, 28, 29] and initially achieved great progress in NLP community. These methods aim to exploit the representation knowledge in the large pretrained models by freezing most parameters of the model and only tuning a small set of parameters, which can achieve comparable or even better performance to full fine-tuning. Several approaches have attempted to apply PETL techniques to CV and V&L domains [11, 33, 40, 43] and achieved promising results on various downstream tasks. Recent works [9, 31, 45] find that different PETL methods have different characteristics and performance on the same downstream task and thus combining multiple PETL techniques may be more effective in improving the performance.

Vision-and-Language Navigation (VLN), which deals with visual, linguistic and robotic action inputs simultaneously, could benefit from the pre-trained large models while suffering from the considerable model size during the downstream tasks fine-tuning. Considering downstream VLN agents are complex enough, full finetuning them with the large pre-trained models for each downstream VLN task becomes expensively, in which case the technique of PETL shows great potential. Unlike most NLP, CV, and V&L tasks, VLN is a dynamic action decision-making task relying on the current environment and previous history knowledge of the chosen actions. Specifically, given the instruction in natural language, the VLN agent perceives a new visual observation according to the chosen action at the previous timestep and should choose the next action to perform at the current timestep. Thus, how to effectively learn history knowledge is crucial to adapting PETL methods for VLN tasks. Moreover, the cross-modal interaction which plays a vital role in action prediction should be also enhanced during the process of efficient tuning. In addition, our experiments show that directly applying some existing PETL methods to VLN tasks may bring non-trivial performance degeneration.

Considering these reasons, we propose a VLN-specific PETL method named VLN-PETL. Specifically, we design two tailored PETL modules for VLN: Historical Interaction Booster (HIB) and Cross-modal Interaction Booster (CIB). Both these two modules mainly consist of bottleneck layers and multi-head cross-attention layers. HIB enhances the interaction between the observation and the previous historical knowledge in a recurrent pattern. While CIB adopts a two-stream structure to focus on the interaction of cross-modal knowledge. Similar to adapters which inject bottleneck layers into transformer blocks for efficient tuning, we insert HIB and CIB into the visual encoder and cross-modal encoder separately in the pre-trained model for VLN. During the training process, the original weights of the large pre-trained model are frozen and only weights of these newly injected modules are trained and updated for different downstream VLN tasks. In addition to HIB and CIB, VLN-PETL also adopts vanilla adapters to efficiently tune the language encoder and the LoRA to further improve the parameter-efficient tuning's performance as previous work declared [9, 31, 45] for downstream VLN tasks.

We conduct extensive experiments on four mainstream VLN tasks: R2R [2], REVERIE [36], NDH [41], and RxR [16]. The results show that VLN-PETL not only surpasses other PETL methods with promising margins but also achieves comparable or even better performance compared to full fine-tuning, especially on R2R ($\uparrow$ 1.3% SR on validation unseen set, updating only 2.82% params, see Figure 1) and on NDH ($\uparrow$ 1.08 GP on test unseen set which achieves the top position in the leaderboard). We also conduct ablation studies to evaluate the contribution of each component of VLN-PETL and validate the superiority of HIB and CIB to counterpart PETL methods.

In summary, our contributions are as follows: (1) We present the first study that explores Parameter-Efficient Transfer Learning (PETL) techniques for Vision-and-Language Navigation (VLN) tasks; (2) We propose a VLN-specific PETL method named VLN-PETL, which incorporates existing PETL methods with two tailored PETL modules for VLN tasks: Historical Interaction Booster (HIB) and Cross-modal Interaction Booster (CIB); (3) Extensive experiments on four VLN downstream tasks demonstrate the effectiveness of our proposed VLN-PETL, which outperforms other PETL methods and keep competitive to full fine-tuning with much fewer trainable parameters.

2. Related work

Vision-and-Language Navigation In the past few years, VLN has received great attention and many methods have been proposed [12, 13, 21, 22, 34, 35]. Early works were mainly based on the encoder-decoder frameworks [2, 6, 26]. While subsequent works approached VLN research in a variety of ways, such as data augmentation [6, 19, 23] to improve the robustness of the agent, progress monitoring [26, 42, 46] to estimate the completeness of instruction-following, and back-tracking [24, 27] to help the agent learn to decide when to perform backtracking depending on the state of the agent. Recently, BERT-based pre-training methods have significantly improved the agents' performance in VLN tasks [7, 20]. These methods follow the pretraining-and-finetuning paradigm, which first pre-trains a vision-and-language model on a great many text-image pairs of instructions and trajectories with specifically designed proxy tasks, and then fully finetunes the pre-trained model for every downstream VLN task. VLNBERT [30] first utilizes this paradigm to solve different downstream VLN tasks, which introduces an extra proxy task of scoring path-instruction pairs in addition to the Masked Language Modeling task. PREVALENT [8] introduces a single-step action prediction proxy task, aiming to learn action-oriented generic visio-linguistic representation. HOP [37] and HOP+ [38] exploits past observations to enhance the learning of temporal order modeling and historical information by introducing three VLN-specific proxy tasks. HAMT [5] encodes past panoramic observations as historical information explicitly.

Though these pretraining-and-finetuning methods have attempted to utilize vital historical knowledge for action prediction, the exploration is still limited due to the gap between pretraining and finetuning. Meanwhile, these methods face the same challenge in the fine-tuning stage: oversized parameters for efficient tuning. Specifically, all parameters of the full pre-trained model will be trained and stored for each downstream VLN task. This may hinder the application of VLN in real-world scenarios since realistic robots will have difficulty in training and storing such huge parameters for every new task. Recent works of Parameter-Efficient Transfer Learning (PETL) show great potential in solving this problem, which freeze most parameters in the large pre-trained model while only training and storing minimal parameters for every downstream task. Thus, we propose the first work that studies and applies PETL methods to VLN named VLN-PETL.

Parameter-Efficient Transfer Learning Recently, with the rapid increase of pre-trained models' size, how to efficiently tune the large pre-trained models has received great attention and Parameter-Efficient Transfer Learning (PETL) has become a popular research area. One category of PETL methods adds new parameters into the pretrained model and only trains these parameters. For example, Adapter [14] introduces bottleneck layers after attention layers and feed-forward layers in the transformer block. LoRA [15] injects trainable low-rank decomposition matrices into linear projection layers to approximate the update of large amount of parameters. Prompt Tuning [18] prepends trainable prompts to the model's input. Another kind of PETL method does not add new parameters and selects a subset of the pretrained model's parameters to update, such as BitFit [3], which only trains the bias term in the model. Recent works [9, 31, 45] find that incorporating different PETL methods as sub-modules may help improve the integrated performance for different downstream tasks in NLP and CV.

However, VLN is a dynamic task of action prediction relying on both the current observations and previous action decisions, which is more challenging than other static NLP, CV and V&L tasks. In other words, the previous decision of action will influence the current environment observed by the agent as well as the choice of the next action to perform. Thus, it is important to effectively utilize the historical knowledge of the previous trajectory when applying PETL methods to VLN. Furthermore, the cross-modal interactions of the language and vision also have a great impact on action predictions, which should be enhanced especially when most parameters of the pre-trained model are frozen for efficient tuning. Therefore, we specifically design two PETL modules for VLN tasks to enhance the history knowledge interactions and cross-modal knowledge interactions, namely Historical Interaction Booster (HIB) and Cross-modal Interaction Booster (CIB). In addition, VLN-PETL also incorporates vanilla adapters to efficiently tune the language encoder and LoRA to further improve the performance.

3. Preliminaries

3.1. Problem Definition of VLN

Given a natural language instruction $\mathcal{I}$ and the current node on the connectivity graph of the environment, the VLN agent should predict an action a_t at each time step t. Specifically, at the time step t, the agent receives a panoramic view of the surrounding environment as the visual observation $\mathcal{O}_t$. $\mathcal{O}_t$ consists of N single-view images and can be represented as a set of features: $\mathcal{O}_t = \{(v_n^o; a_n^o)\}$, where v_n^o represents the feature of the n-th single-view image and a_n^o represents the corresponding angle feature. The history $\mathcal{H}_t = \{(\mathcal{O}_i; a_i^h)\}$ that consists of every observation $\mathcal{O}_i$ and the performed action a_i^h (*i.e.* the turned angles) at each time step i before t also plays a vital role in the agent's action prediction. Based on $\mathcal{I}$, $\mathcal{O}_t$ and $\mathcal{H}_t$, the VLN agent predicts the next action a_t at each time step t to navigate to the target goal until the special `[STOP]` action is selected, or the step reaches the maximum length. Some VLN tasks such as REVERIE [36] additional require the agent to return a target object location, while some others such as the NDH [41] use dialogue instructions.

3.2. PETL Methods

Adapter inserts trainable bottleneck layers after the multi-head attention layers or feed-forward layers in the transformer blocks. The bottleneck layer of an adapter consists of a linear down-projection with $\boldsymbol{W}_{\text{down}} \in \mathbb{R}^{D_{\text{hidden}} \times D_{\text{mid}}}$, a non-linear activation function $\sigma(\cdot)$ and a linear up-projection $\boldsymbol{W}_{\text{up}} \in \mathbb{R}^{D_{\text{mid}} \times D_{\text{hidden}}}$. Given the input feature $\boldsymbol{f}_{\text{in}}$, the adapter first projects $\boldsymbol{f}_{\text{in}}$ into the D_{mid} bottleneck dimension and then recovers it back into D_{hidden} dimension as:

$$\boldsymbol{f}_{\text{out}} = \boldsymbol{W}_{\text{up}}^{\intercal} \sigma(\boldsymbol{W}_{\text{down}}^{\intercal} \boldsymbol{f}_{\text{in}}). \tag{1}$$

Bias terms are omitted for brevity. The parameters of layer normalization are usually tuned together with the adapter. Besides, the adapter can be inserted into the transformer layers in a sequential manner or in a parallel manner, while the latter one is proved superior to the former one as in [9].

LoRA injects trainable low-rank decomposition matrices to represent the weight updates of the frozen parameters in the transformer's linear projection layers. Specifically, for a weight matrix $\boldsymbol{W} \in \mathbb{R}^{D_{\text{hidden}} \times D_{\text{hidden}}}$ in the pre-trained model, the weight update $\Delta \boldsymbol{W} \in \mathbb{R}^{D_{\text{hidden}} \times D_{\text{hidden}}}$ is approximated by two low-rank matrices $\boldsymbol{W}_{\text{down}} \in \mathbb{R}^{D_{\text{hidden}} \times D_{\text{mid}}}$ and $\boldsymbol{W}_{\text{up}} \in \mathbb{R}^{D_{\text{mid}} \times D_{\text{hidden}}}$ as follow:

$$\boldsymbol{W} + \Delta \boldsymbol{W} = \boldsymbol{W} + \boldsymbol{W}_{\text{down}} \boldsymbol{W}_{\text{up}}, \tag{2}$$

and the forward pass of LoRA can be formulated as:

$$\boldsymbol{f}_{\text{out}} = (\boldsymbol{W}^{\intercal} + \gamma \boldsymbol{W}_{\text{up}}^{\intercal} \boldsymbol{W}_{\text{down}}^{\intercal}) \boldsymbol{f}_{\text{in}}. \tag{3}$$

where γ is a fixed scalar hyperparameter for scaling.

Prompt Tuning prepends a sequence of randomly initialized continuous prompts $\boldsymbol{f}_{\text{prompt}}$ into the input feature $\boldsymbol{f}_{\text{in}}$. During training, only these prompts are optimized by updating a learnable projection matrix $\boldsymbol{W}_{\text{prompt}} \in \mathbb{R}^{1 \times D_{\text{hidden}}}$. The forward pass can be formulated as:

$$\boldsymbol{f}_{\text{prompt}} = \boldsymbol{W}_{\text{prompt}}^{\intercal} \boldsymbol{X}, \tag{4}$$

$$\boldsymbol{f}_{\text{out}} = \text{TRM}([\boldsymbol{f}_{\text{prompt}}, \boldsymbol{f}_{\text{in}}]). \tag{5}$$

where $\boldsymbol{X}$ represents discrete prompt tokens, $[\cdot]$ represents concatenation, $\text{TRM}(\cdot)$ represents the transformer block.

BitFit does not introduce new parameters or inputs for tuning. It only tunes the bias terms in the pretrained models, which shows promising performance in some NLP tasks.

4. Method

4.1. Baseline Model

We use a large pre-trained model of HAMT [5] as our baseline, which is pre-trained on plentiful text-image pairs of instructions and the corresponding trajectories. As illustrated in Figure 2, HAMT adopts a two-stream architecture, which consists of a language encoder and a vision encoder to extract single-modal features, and a cross-modal encoder to fuse multi-modal features for action prediction.

Language Encoder Following the practice of BERT, the language encoder first embeds the instruction $\mathcal{I}$ into language embedding E_x by summing the word embedding, position embedding, and type embedding of each word x_i in the instruction. Then, E_x is passed through N_L transformer blocks which consist of a multi-head self-attention layer and feed-forward layer to generate the language feature $\boldsymbol{f}_x$.

Vision Encoder The vision encoder mainly consists of observation encoding and history encoding. At time step t, the panoramic image feature v_t^o and the corresponding angle feature a_t^o are projected into E_v^o and E_a^o followed by the layer normalization. Then, E_v^o and E_a^o are summed up as the current observation feature $\boldsymbol{o}_t$. Meanwhile, the panoramic image feature v_{t-1}^h and the corresponding turned angle feature a_{t-1}^h of the previous time step are taken as the input for history encoding. Similarly, v_{t-1}^h and a_{t-1}^h are first projected into E_v^h and E_a^h. Then, E_v^h and E_a^h are summed up and passed through N_H transformer blocks to generate h_t, which is appended into the tail of $\langle h_1, ..., h_{t-1}\rangle$ as the current time step's history feature $\boldsymbol{h}_t$. Indeed, the history encoding only re-encodes the previous time step's observation statically without any other interactions. Thus, this manner of history feature's generation and update may neglect the temporal knowledge embodied in the navigation trajectory.

Cross-modal Encoder At the time step t, the observation feature $\boldsymbol{o}_t$ and the history feature $\boldsymbol{h}_t$ are first concatenated as the visual feature $\boldsymbol{f}_v$. Then, $\boldsymbol{f}_x$ and $\boldsymbol{f}_v$ are passed through N_C cross-modal transformer blocks that consist of two-stream multi-head cross-attention layers, multi-head self-attention layers, and feed-forward layers to generate cross-modal features for action prediction.

4.2. VLN-PETL

Not all the aforementioned PETL methods perform well in the complicated VLN tasks, such as BitFit and Prompt Tuning, which bring performance degeneration to VLN tasks (see Sec. 6 for evaluation and explanation). Thus, as shown in Figure 2, we only incorporate the adapter and LoRA as the PETL components of our integrated VLN-PETL. Furthermore, two block-level PETL modules are specially designed for VLN tasks considering the unique characteristics of VLN tasks and parameter efficiency, namely Historical Interaction Booster (HIB) and Cross-modal Interaction Booster (CIB). Based on the bottleneck structure of the adapter, these two modules respectively strengthen the historical interaction and cross-modal interaction by incorporating the multi-head cross-attention mechanism and gating mechanism.

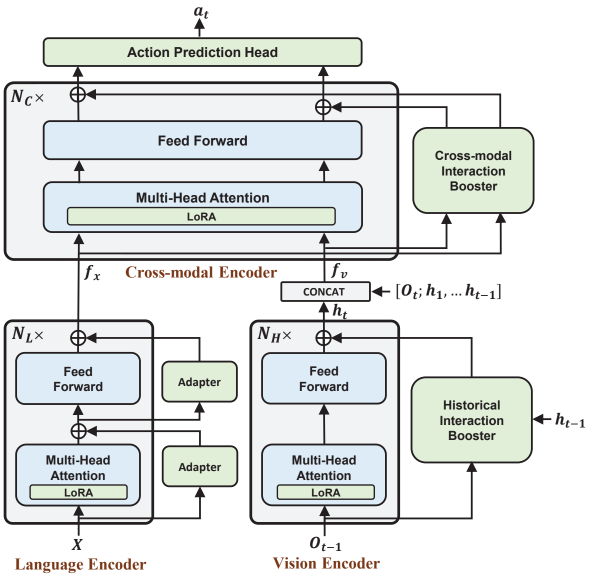


Figure 2: Illustration of the framework of our proposed VLN-PETL. The pre-trained model of HAMT mainly consists of a language encoder, a vision encoder and a cross-modal encoder. The blue color denotes the frozen parameters in the pre-trained model and the green color denotes the trainable parameters of injected PETL modules.

Language Encoder Adapter As shown in Figure 3a, the Language Encoder Adapter (LEA) is inserted into the multi-head self-attention layers and feed-forward layers in parallel. Concretely, for the l-th transformer block in the language encoder, the input feature $\boldsymbol{f}_x^{l-1}$ is first passed through the adapter and summed with the output feature of the multi-head self-attention layer as $\boldsymbol{f}_{\text{att}}$:

$$\boldsymbol{f}_{\text{att}} = \text{LN}\big(\text{MSA}(\boldsymbol{f}_x^{l-1}) + \text{ADAPTER}(\boldsymbol{f}_x^{l-1})\big), \tag{6}$$

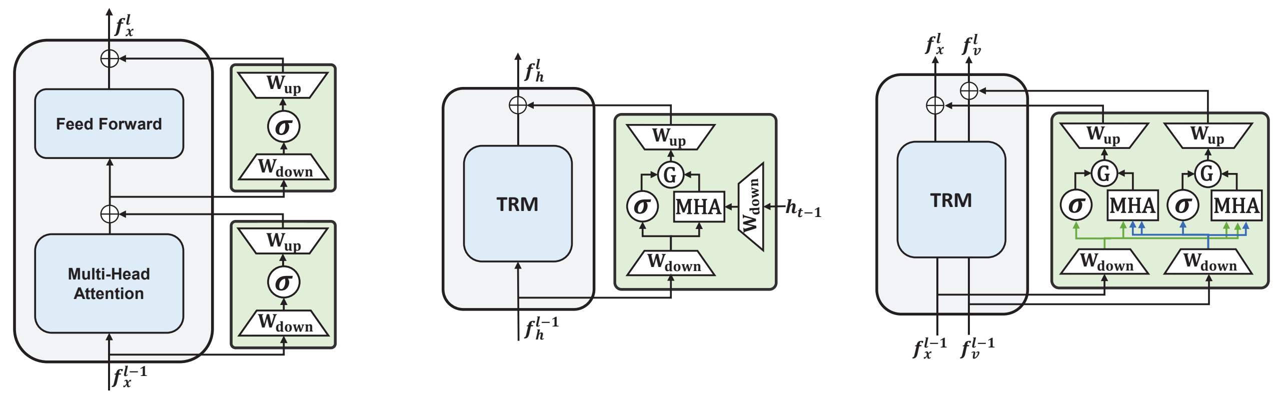
where $\text{MSA}(\cdot)$ represents the multi-head self-attention layer, $\text{ADAPTER}(\cdot)$ represents the adapter block shown in Eq.1 and $\text{LN}(\cdot)$ represents the layer normalization. Similarly, another adapter is inserted into the feed-forward layer which takes $\boldsymbol{f}_{\text{att}}$ as input:

$$\boldsymbol{f}_{\text{ffn}} = \text{LN}\big(\text{FFN}(\boldsymbol{f}_{\text{att}}) + \text{ADAPTER}(\boldsymbol{f}_{\text{att}})\big), \tag{7}$$

where $\text{FFN}(\cdot)$ represents the feed-forward layer. $\boldsymbol{f}_{\text{ffn}}$ is used as the final output feature $\boldsymbol{f}_x^l$ of the l-th transformer block:

$$\boldsymbol{f}_x^l = \boldsymbol{f}_{\text{ffn}}. \tag{8}$$

Historical Interaction Booster As shown in Figure 3b, the Historical Interaction Booster (HIB) adopts the multi-head



(a) Language Encoder Adapter (LEA). (b) Historical Interaction Booster(HIB). (c) Cross-modal Interaction Booster(CIB).

Figure 3: Detailed components of VLN-PETL. TRM represents the transformer block, MHA represents the multi-head attention layer, σ represents the activation layer and G represents the learnable gate.

cross-attention mechanism to enhance the historical interaction between the observation feature and the history feature at each timestep t in a recurrent pattern. Specifically, for the l-th transformer block in the vision encoder, the input observation feature $\boldsymbol{f}_h^{l-1}$ and the previous history feature $\boldsymbol{h}_{t-1}$ are first downsampled into $\boldsymbol{f}_{\text{down}}$ and $\boldsymbol{h}_{\text{down}}$ with D_{mid} dimension by projection matrices $\boldsymbol{W}_{\text{down\_f}}$ and $\boldsymbol{W}_{\text{down\_h}}$:

$$\boldsymbol{f}_{\text{down}} = \boldsymbol{W}_{\text{down\_f}}^{\intercal}\boldsymbol{f}_h^{l-1}, \tag{9}$$

$$\boldsymbol{h}_{\text{down}} = \boldsymbol{W}_{\text{down\_h}}^{\intercal}\boldsymbol{h}_{t-1}, \tag{10}$$

Then, the history knowledge is encoded into the observation feature by the multi-head cross-attention between $\boldsymbol{f}_{\text{down}}$ and $\boldsymbol{h}_{\text{down}}$ followed by a learnable gate α:

$$\boldsymbol{f}'_{\text{down}} = \text{ReLU}(\boldsymbol{f}_{\text{down}}), \tag{11}$$

$$\boldsymbol{f}_{\text{cross}} = \text{MHA}(\boldsymbol{f}_{\text{down}}, \boldsymbol{h}_{\text{down}}), \tag{12}$$

$$\alpha = \text{Sigmoid}(\frac{\theta}{T}), \tag{13}$$

$$\boldsymbol{f}_{\text{v\_h}} = \alpha * \boldsymbol{f}'_{\text{down}} + (1-\alpha) * \boldsymbol{f}_{\text{cross}}, \tag{14}$$

where $\text{MHA}(\cdot)$ represents the multi-head cross-attention layer of which the query is $\boldsymbol{f}_{\text{down}}$ while the key and value are $\boldsymbol{h}_{\text{down}}$, θ is a learnable scalar initialized by zero and T is fixed as 0.1 representing the temperature hyperparameter. Next, the attended visual-and-historical feature $\boldsymbol{f}_{\text{v\_h}}$ is upsampled into D_{hidden} dimension and summed with the original output feature $\hat{\boldsymbol{f}}_t^l$ of the l-th transformer block:

$$\hat{\boldsymbol{f}}_h^l = \text{TRM}(\boldsymbol{f}_h^{l-1}), \tag{15}$$

$$\boldsymbol{f}_h^l = \text{LN}(\hat{\boldsymbol{f}}_h^l + \boldsymbol{W}_{\text{up}}^{\intercal}\boldsymbol{f}_{\text{v\_h}}), \tag{16}$$

where $\text{TRM}(\cdot)$ represents the transformer block in the vision encoder. The final output history feature h_t at the current timestep t is obtained as follow:

$$h_t = \boldsymbol{f}_h^L, \tag{17}$$

where L represents the number of transformer blocks.

Cross-modal Interaction Booster As shown in Figure 3c, to enhance the interaction between language and visual modalities, Cross-modal Interaction Booster (CIB) adopts a two-stream multi-head cross-attention mechanism. To be specific, for the l-th transformer block in the cross-modal encoder, the input language feature $\boldsymbol{f}_x^{l-1}$ and the visual feature $\boldsymbol{f}_v^{l-1}$ are first downsampled into $\boldsymbol{f}_{\text{down\_x}}$ and $\boldsymbol{f}_{\text{down\_v}}$ with D_{mid} dimension by projection matrices $\boldsymbol{W}_{\text{down\_x}}$ and $\boldsymbol{W}_{\text{down\_v}}$ as Eq.9 and Eq.10. Then, a two-stream multi-head cross-attention is implemented by exchanging the query for the key and value as follows:

$$\boldsymbol{f}_{\text{cross\_x}} = \text{MHA}(\boldsymbol{f}_{\text{down\_x}}, \boldsymbol{f}_{\text{down\_v}}), \tag{18}$$

$$\boldsymbol{f}_{\text{cross\_v}} = \text{MHA}(\boldsymbol{f}_{\text{down\_v}}, \boldsymbol{f}_{\text{down\_x}}), \tag{19}$$

Two learnable gates α_x and α_v are used to obtain cross-attended language feature $\boldsymbol{f}_{\text{x\_v}}$ and visual feature $\boldsymbol{f}_{\text{v\_x}}$:

$$\boldsymbol{f}_{\text{x\_v}} = \alpha_x * \boldsymbol{f}'_{\text{down\_x}} + (1-\alpha_x) * \boldsymbol{f}_{\text{cross\_x}}, \tag{20}$$

$$\boldsymbol{f}_{\text{v\_x}} = \alpha_v * \boldsymbol{f}'_{\text{down\_v}} + (1-\alpha_v) * \boldsymbol{f}_{\text{cross\_v}}, \tag{21}$$

where $\boldsymbol{f}'_{\text{down\_x}}$ and $\boldsymbol{f}'_{\text{down\_v}}$ are obtained by passing $\boldsymbol{f}_{\text{down\_x}}$ and $\boldsymbol{f}_{\text{down\_v}}$ through ReLU layer as Eq.11. At last, $\boldsymbol{f}_{\text{x\_v}}$ and $\boldsymbol{f}_{\text{v\_x}}$ are respectively upsampled into D_{hidden} dimension and summed with the original output language feature $\hat{\boldsymbol{f}}_x^l$ and visual feature $\hat{\boldsymbol{f}}_v^l$ as the final output feature $\boldsymbol{f}_x^l$ and $\boldsymbol{f}_v^l$ of the l-th transformer block:

$$\hat{\boldsymbol{f}}_x^l, \hat{\boldsymbol{f}}_v^l = \text{TRM}(\boldsymbol{f}_x^{l-1}, \boldsymbol{f}_v^{l-1}), \tag{22}$$

$$\boldsymbol{f}_x^l = \text{LN}(\hat{\boldsymbol{f}}_x^l + \boldsymbol{W}_{\text{up\_x}}^{\intercal}\boldsymbol{f}_{\text{x\_v}}), \tag{23}$$

$$\boldsymbol{f}_v^l = \text{LN}(\hat{\boldsymbol{f}}_v^l + \boldsymbol{W}_{\text{up\_v}}^{\intercal}\boldsymbol{f}_{\text{v\_x}}). \tag{24}$$

where $\text{TRM}(\cdot)$ represents the transformer block in the cross-modal encoder.

Incorporating LoRA in VLN-PETL The multi-head attention layers account for a large portion of parameters in

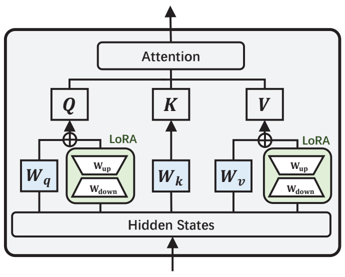


Figure 4: Illustration of injecting LoRA in the Multi-head Attention layer.

the transformer. Thus, as shown in Figure 4, VLN-PETL incorporates LoRA as an independent PETL component in addition to LEA, HIB and CIB to further improve the performance on downstream VLN tasks. Specifically, we globally inject LoRA layers into query matrices $\boldsymbol{W}_Q$ and value matrices $\boldsymbol{W}_V$ of all multi-head attention layers in the pre-trained model. Given the input feature $\boldsymbol{f}_{\text{in\_q}}$ and $\boldsymbol{f}_{\text{in\_v}}$ for linear projection in the multi-head attention layer, the output feature $\boldsymbol{Q}$ and $\boldsymbol{V}$ can be computed as follow:

$$\boldsymbol{Q} = (\boldsymbol{W}_Q^\intercal + \gamma \boldsymbol{W}_{\text{up\_q}}^\intercal \boldsymbol{W}_{\text{down\_q}}^\intercal)\boldsymbol{f}_{\text{in\_q}}, \tag{25}$$

$$\boldsymbol{V} = (\boldsymbol{W}_V^\intercal + \gamma \boldsymbol{W}_{\text{up\_v}}^\intercal \boldsymbol{W}_{\text{down\_v}}^\intercal)\boldsymbol{f}_{\text{in\_v}}. \tag{26}$$

5. Experimental Setup

5.1. Downstream tasks

To comprehensively evaluate PETL methods for VLN, we conduct experiments on four downstream tasks: Room-to-Room (R2R) [2], REVERIE [36], NDH [41] and Room-across-Room (RxR) [16]. These downstream tasks evaluate the agent from different views: (1) R2R and RxR require the agents to follow detailed instructions to navigate from one room to another; (2) REVERIE gives a concise, high-level instruction referring to a remote object, which focuses on grounding remote target objects; (3) NDH requires an agent to reach target regions based on the dialog history, which contains multiple question-and-answer interactions between the agent and an oracle.

5.2. Evaluation metrics

We follow previous work and adopt the most commonly used metrics for evaluating VLN agents as follows: **`TL`** (Trajectory Length) measures the average length of all the predicted navigation trajectories in meters. **`NE`** (Navigation Error) is the mean the average distance in meters between the agent's final location and the target location. **`SR`** (Success Rate) measures the ratio of successful tasks, of which the agent's stop location is less than 3 meters away from the target location. **`SPL`** (Success weighted by Path Length [1]) trades-off SR (Success Rate) against TL (Trajectory Length), which measures both the accuracy and efficiency of navigation. **`OSR`** (Oracle Success Rate) measures the ratio of tasks of which one of its trajectory viewpoints can observe the target object within 3 meters. **`RGS`** (Remote Grounding Success rate) measures the ratio of tasks that successfully locate the target object. **`RGSPL`** (RGS weighted by Path Length) is RGS weighted by Path Length. **`GP`** (Goal Progress) measures the average progress of the agent towards the target. **`nDTW`** (Normalized Dynamic Time Warping) penalizes deviations from the reference path. **`sDTW`** (Success weighted by normalized Dynamic TimeWarping) constrains nDTW to only successful episodes and effectively captures both success and fidelity.

5.3. Implementation details

We choose existing PETL methods of BitFit, Prompt Tuning, Adapter and LoRA for comparison with our integrated VLN-PETL. We use the same learning rate of $1e-4$, and AdamW [25] optimizer for all PETL methods. The batch size is set as 4 for REVERIE and 8 for the other three VLN tasks. For Prompt Tuning, we respectively add 20 prompt tokens in front of the inputs of the language encoder and vision encoder. For the setting of both Adapter and LoRA, the bottleneck dimension D_{dim} is set as 64, which brings comparative parameters for a fair comparison with VLN-PETL. While for VLN-PETL, the bottleneck dimensions D_{dim} of the Language Encoder Adapter (LEA), History Interaction Booster (HIB), and Cross-modal Interaction Booster (CIB) is set as 64 while the bottleneck dimension of the incorporated LoRA is set as 8 for REVERIE while 16 for other VLN tasks. The attention heads number of both HIB and CIB is set as 4. For all PETL methods, the prediction heads of the pre-trained model are also trained. For a fair comparison, we follow HAMT [5] to use Reinforcement Learning (RL) and Imitation Learning (IL).

6. Experimental Results

6.1. Comparison of PETL methods for VLN

As shown in Table 1-4, we compare our proposed VLN-PETL with finetuning, Bitfit [3], Prompt Tuning [18], LoRA [15], and Adapter [14] in performance and trainable parameter amounts on different VLN tasks.

We can see that Prompt Tuning with the least updated parameters works poorly for all VLN downstream tasks. One possible reason may be the limited trainable parameters. More importantly, the training instability of Prompt Tuning as previous work declared should also be responsible for the poor performance. In fact, training VLN agents itself is not always stable and easy, where reinforcement learning plays a vital role. BitFit surpasses Prompt Tuning with a non-trivial margin on all tasks with comparable amounts of trainable parameters. However, the performance of BitFit still falls far from finetuning, LoRA, and Adapter on all VLN tasks especially on RxR in high demand for lan-

Methods	Updated Params(%)	Validation Seen TL	Validation Seen NE ↓	Validation Seen SR ↑	Validation Seen SPL ↑	Validation Unseen TL	Validation Unseen NE ↓	Validation Unseen SR ↑	Validation Unseen SPL↑	Test Unseen TL	Test Unseen NE ↓	Test Unseen SR ↑	Test Unseen SPL ↑
Fine-Tuning	100	11.48	2.94	72.67	69.17	11.62	3.64	64.24	59.25	12.20	4.09	63.20	58.55
BitFit [3]	0.46	11.61	3.78	63.47	60.35	12.22	4.18	59.17	54.67	12.96	4.63	57.15	53.03
Prompt Tuning [18]	0.37	10.67	4.24	61.02	58.59	11.14	4.63	56.49	52.32	11.60	4.88	54.47	50.91
LoRA [15]	3.02	11.73	3.14	70.13	66.00	12.25	3.84	63.60	57.59	12.99	4.15	61.44	55.96
Adapter [14]	3.08	11.70	3.34	67.38	64.42	12.66	4.00	63.01	57.42	13.19	4.27	60.69	55.88
VLN-PETL(ours)	2.82	11.39	**2.93**	**72.28**	**68.50**	11.52	**3.53**	**65.47**	**60.01**	12.30	**4.10**	**63.22**	**58.25**

Table 1: Performance of PETL methods on R2R. For each method, we report the percentage of trainable parameters compared to full fine-tuning. **Bold** and underline denote the best and runner-up results. SPL is the main metric.

Methods	Updated	REVERIE Validation Seen						REVERIE Validation Unseen						REVERIE Test Unseen					
		Navigation				RGS↑	RGSPL↑	Navigation				RGS↑	RGSPL↑	Navigation				RGS↑	RGSPL↑
	Params(%)	SR↑	OSR↑	SPL↑	TL			SR↑	OSR↑	SPL↑	TL			SR↑	OSR↑	SPL↑	TL		
Fine-Tuning	100	46.73	52.35	42.75	13.37	30.64	27.91	32.63	37.82	28.92	15.66	18.66	16.06	33.09	37.82	27.02	12.83	15.04	13.32
BitFit [3]	0.80	33.52	37.81	31.84	11.45	19.96	18.93	24.65	27.46	21.34	12.36	10.85	9.43	21.50	24.95	18.85	12.53	9.87	8.62
Prompt Tuning [18]	0.71	25.86	33.17	23.21	12.15	8.29	7.44	19.94	25.08	17.75	12.43	5.82	5.07	19.95	24.24	17.94	11.61	5.51	4.88
LoRA [15]	3.33	42.30	46.24	38.63	12.89	29.87	27.45	29.42	34.28	26.17	15.96	15.25	13.45	**32.12**	37.00	**26.86**	14.93	14.94	12.76
Adapter [14]	3.39	40.76	45.05	37.43	13.75	27.13	24.62	29.48	32.83	26.62	14.59	16.05	14.21	29.20	32.31	24.78	14.96	14.51	12.48
VLN-PETL(ours)	2.81	**45.96**	51.23	**42.60**	12.86	**29.94**	**27.61**	**31.81**	37.03	**27.67**	14.47	**18.26**	**15.96**	30.83	36.06	26.73	14.00	**15.13**	**13.03**

Table 2: Performance of PETL methods on REVERIE. SPL is the main metric for its navigation task, and RGSPL is the main metric for the object grounding task.

Methods	Updated Params(%)	Val Seen	Val Unseen	Test Unseen
Fine-Tuning	100	7.69	5.16	5.05
BitFit [3]	0.46	6.68	3.77	4.03
Prompt Tuning [18]	0.37	4.71	3.26	2.86
LoRA [15]	3.02	6.83	5.16	5.91
Adapter [14]	3.08	5.29	5.30	4.72
VLN-PETL(ours)	2.82	**7.76**	**5.69**	**6.13**

Table 3: Performance on NDH measured by Goal Progress.

guage understanding with much longer instructions. Only tuning bias terms may have difficulties in handling these complex VLN tasks. Thus, we believe that Prompt Tuning and BitFit are not applicable for efficiently tuning large pre-trained models for challenging VLN tasks. While LoRA and Adapter not only have comparative amounts of trainable parameters but also have comparable performances on all VLN tasks. These two methods further shrink the performance gap with finetuning, which are potential to effectively tune VLN pretrained models.

As for VLN-PETL, though it has fewer parameters than LoRA and Adapter, VLN-PETL still surpasses LoRA and Adapter on most evaluation metrics in all four downstream VLN tasks. Furthermore, only VLN-PETL maintains competitive performances compared to fine-tuning and even outperforms fine-tuning on several evaluation metrics. As shown in Table 3, it is worth mentioning that VLN-PETL outperforms full fine-tuning on all dataset splits in the NDH task, and achieves the top position on the public leaderboard[1]. These promising results demonstrate the effectiveness of our proposed VLN-PETL for efficiently tune large pre-trained models for VLN tasks.

[1]https://eval.ai/web/challenges/challenge-page/463/leaderboard/1292 (01/03/2023)

6.2. Ablation Study

Contribution of VLN-PETL components As shown in Table 5, to evaluate the contribution of LEA, HIB and CIB, we choose REVERIE and RxR which are more challenging VLN tasks to conduct ablation studies. REVERIE not only measures the agent's ability in navigation but also in locating the target object, while RxR has much longer instructions requiring comprehensive language understanding. We also report the results of only tuning the prediction head for comparison. We find that LEA has a competitive performance compared to HIB, and only tuning either LEA or HIB outperforms the head tuning with a nontrivial margin. While CIB contributes much more inefficiently tuning the VLN model, which improves the performance with a larger margin on both REVERIE and RxR. This result indicates that language understanding and vision understanding with history knowledge contribute comparably to the action prediction for VLN agents, while the cross-modal interaction plays more importantly in this process. By combining all these three components, the VLN agent achieves a promising performance and outperforms other PETL methods, especially on the RGSPL metric, which measures the agent's ability to locate the target object.

Superiority of HIB and CIB As shown in Table 6, to validate the effectiveness of HIB and CIB, we compare the performance of HIB and CIB with their counterparts of Adapter, by respectively replacing HIB and CIB by History Encoder Adapter (HEA) and Cross-modal Encoder Adapter (CEA) which are similar to Language Encoder Adapter. Due to the enhancement of historical knowledge learning, HIB surpasses HEA on seen set by a large margin. On the unseen set, HIB falls behind HEA with a trivial margin on the SPL metric while outperforming HEA on the

Methods	Updated Params(%)	RxR Validation Seen SR↑	SPL ↑	nDTW ↑	sDTW ↑	RxR Validation Unseen SR↑	SPL ↑	nDTW ↑	sDTW↑
Fine-Tuning	100	64.93	61.28	69.26	55.92	57.88	54.18	64.52	49.44
BitFit [3]	0.28	35.69	33.58	49.37	29.81	36.63	34.34	50.47	30.49
Prompt Tuning [18]	0.23	27.92	25.98	42.93	22.97	29.95	27.72	44.94	24.67
LoRA [15]	1.86	55.66	52.41	63.20	47.31	54.53	51.14	63.23	46.94
Adapter [14]	1.90	<u>58.52</u>	<u>54.96</u>	<u>65.14</u>	<u>50.23</u>	<u>55.19</u>	<u>51.44</u>	<u>63.56</u>	<u>47.32</u>
VLN-PETL(ours)	1.67	**60.48**	**56.77**	**65.74**	**51.67**	**57.95**	**54.16**	**64.94**	**49.70**

Table 4: Performance on RxR using English instructions. nDTW is the main metric for the RxR task.

	Components LEA	HIB	CIB	LoRA	REVERIE Val Unseen SPL↑	RGSPL↑	RxR Val Unseen SR↑	SPL↑
1	×	×	×	×	15.19	4.53	28.66	26.72
2	✓	×	×	×	18.21	9.73	43.45	40.68
3	×	✓	×	×	18.49	9.32	41.04	38.46
4	×	×	✓	×	21.62	12.86	52.01	48.82
5	✓	✓	×	×	21.86	11.75	45.08	42.11
6	✓	×	✓	×	25.00	14.34	52.26	49.26
7	×	✓	✓	×	25.55	14.89	54.26	50.71
8	✓	✓	✓	×	26.51	15.29	56.04	52.79
9	✓	✓	✓	✓	**27.67**	**15.96**	**57.95**	**54.16**

Table 5: Ablation of different components in VLN-PETL on REVERIE Unseen set and RxR Unseen set.

Methods	Validation Seen SPL↑	RGSPL↑	Validation Unseen SPL↑	RGSPL↑
HEA	25.74	12.51	18.98	8.39
HIB	28.55	16.95	18.49	9.32
CEA	31.44	17.55	20.77	10.14
CIB	38.99	25.46	21.62	12.86

Table 6: Performance comparison of HIB and CIB with their counterparts of Adapter on REVERIE val set.

Methods	Validation Seen SPL↑	RGSPL↑	Validation Unseen SPL↑	RGSPL↑
VLN-PETL	42.60	27.61	27.67	15.96
w/o LoRA	41.34	27.12	26.51	15.29

Table 7: Ablation of LoRA's effect on REVERIE val set.

T	0.01	0.1	1	10
SPL↑	25.60	27.67	**27.81**	25.00
RGSPL↑	14.41	**15.96**	15.36	14.06

α	0.5	learnable
SPL↑	27.51	**27.67**
RGSPL↑	14.50	**15.96**

Table 8: Ablation of T and α in the gates of HIB and CIB on REVERIE Val Unseen set.

RGSPL metric with a large margin. This is probably because the input for history encoding is a panoramic view image rather than a single-view image of the front view, where HIB tends to learn more knowledge about the fine-grained object rather than the trajectory. As for CIB and CEA, CIB surpasses CEA on all metrics and all sets with a large margin, which shows the superiority of CIB.

The Effect of LoRA As shown in Table 7, we find that when removing LoRA, the performance of VLN-PETL has a slight drop on all main metrics on both REVERIE seen and unseen splits. Besides, the decrease on RGSPL metric is less than that on SPL metric, which indicates LoRA's effect on the VLN agent's ability to locate objects is smaller than that of navigation during efficient tuning.

Hyper-parameters in Gates. As shown in Table 8, we conduct ablation studies on T and α in the gates of HIB and CIB. The performances are similarly high when T is set as 0.1 or 1. We set T as 0.1 due to its higher score on RGSPL. We also compare the results of fixing α as 0.5 and using the learnable gate. We can see that the learnable gate α surpasses the fixed α with a large margin on RGSPL.

7. Limitations and Future work

Though VLN-PETL is proven to be effective on four mainstream VLN tasks of R2R, REVERIE, NDH, and RxR, all these tasks focus on the agent's ability to navigate or ground target object, which has no interactions with the observed objects. Thus, our future work will pay attention to applying PETL methods to other VLN tasks that have interactions with the environment, such as object manipulation.

8. Conclusion

In this paper, we present the first study of applying Parameter-Efficient Transfer Learning (PETL) methods to VLN tasks and propose a VLN-specific PETL method named VLN-PETL. Considering the characteristics of VLN, we specifically design two PETL modules to efficiently tune the large pre-trained model for VLN downstream tasks, namely Historical Interaction Booster (HIB) and Cross-modal Interaction Booster (CIB). Both HIB and HIB mainly consist of bottleneck layers and multi-head attention layers, which respectively enhance the vision encoder's learning of history knowledge and the cross-modal encoder's interactions between the language and vision features during the efficient tuning. In addition, we incorporate the vanilla adapters to efficiently tune the language encoder and the LoRA to further improve the integrated performance. Extensive experiments conducted on four mainstream VLN tasks of R2R, REVERIE, NDH, and RxR show the effectiveness of our proposed VLN-PETL. Furthermore, we conduct ablation studies to evaluate the contribution of VLN-PETL components and validate the superiority of our specifically designed HIB and CIB to their counterpart PETL methods.

References

[1] Peter Anderson, Angel X. Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir Roshan Zamir. On evaluation of embodied navigation agents. *CoRR*, abs/1807.06757, 2018. 6

[2] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian D. Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, pages 3674–3683, 2018. 2, 6

[3] Elad Ben-Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *ArXiv*, abs/2106.10199, 2022. 1, 3, 6, 7, 8

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, pages 1877–1901, 2020. 1

[5] Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. History aware multimodal transformer for vision-and-language navigation. In *NeurIPS*, pages 5834–5847, 2021. 2, 4, 6

[6] 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 *NeurIPS*, pages 3318–3329, 2018. 2

[7] Pierre-Louis Guhur, Makarand Tapaswi, Shizhe Chen, Ivan Laptev, and Cordelia Schmid. Airbert: In-domain pretraining for vision-and-language navigation. In *ICCV*, pages 1634–1643, 2021. 2

[8] Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *CVPR*, pages 13134–13143, 2020. 2

[9] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. In *ICLR*, 2022. 1, 2, 3

[10] Ruidan He, Linlin Liu, Hai Ye, Qingyu Tan, Bosheng Ding, Liying Cheng, Jia-Wei Low, Lidong Bing, and Luo Si. On the effectiveness of adapter-based tuning for pretrained language model adaptation. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *ACL/IJCNLP*, pages 2208–2222, 2021. 1

[11] Xuehai He, Chunyuan Li, Pengchuan Zhang, Jianwei Yang, and Xin Eric Wang. Parameter-efficient fine-tuning for vision transformers. *CoRR*, abs/2203.16329, 2022. 1

[12] Yicong Hong, Zun Wang, Qi Wu, and Stephen Gould. Bridging the gap between learning in discrete and continuous environments for vision-and-language navigation. In *CVPR*, pages 15418–15428, 2022. 2

[13] Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez Opazo, and Stephen Gould. VLN↻BERT: A recurrent vision-and-language BERT for navigation. In *CVPR*, pages 1643–1653, 2021. 2

[14] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *ICML*, pages 2790–2799, 2019. 1, 3, 6, 7, 8

[15] Edward Hu, Yelong Shen, Phil Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022. 1, 3, 6, 7, 8

[16] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *EMNLP*, pages 4392–4412, 2020. 2, 6

[17] Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. Stacked cross attention for image-text matching. In *ECCV*, pages 212–228, 2018. 1

[18] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *EMNLP*, pages 3045–3059, 2021. 3, 6, 7, 8

[19] Jialu Li, Hao Tan, and Mohit Bansal. Envedit: Environment editing for vision-and-language navigation. In *CVPR*, pages 15386–15396, 2022. 2

[20] Xiujun Li, Chunyuan Li, Qiaolin Xia, Yonatan Bisk, Asli Çelikyilmaz, Jianfeng Gao, Noah A. Smith, and Yejin Choi. Robust navigation with language pretraining and stochastic sampling. In *EMNLP*, pages 1494–1499, 2019. 2

[21] Bingqian Lin, Yi Zhu, Zicong Chen, Xiwen Liang, Jian zhuo Liu, and Xiaodan Liang. Adapt: Vision-language navigation with modality-aligned action prompts. In *CVPR*, pages 15375–15385, 2022. 2

[22] Chuang Lin, Yi Jiang, Jianfei Cai, Lizhen Qu, Gholamreza Haffari, and Zehuan Yuan. Multimodal transformer with variable-length memory for vision-and-language navigation. *ArXiv*, abs/2111.05759, 2021. 2

[23] Chong Liu, Fengda Zhu, Xiaojun Chang, Xiaodan Liang, Zongyuan Ge, and Yi-Dong Shen. Vision-language navigation with random environmental mixup. In *ICCV*, pages 1624–1634, 2021. 2

[24] Yonatan Bisk Ari Holtzman Zhe Gan Jingjing Liu Jianfeng Gao Yejin Choi Siddhartha Srinivasa. Liyiming Ke, Xiujun Li. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *CVPR*, pages 6741–6749, 2019. 2

[25] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*. OpenReview.net, 2019. 6

[26] Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. Self-monitoring navigation agent via auxiliary progress estimation. In *ICLR*, 2019. 2

[27] Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. The regretful agent: Heuristic-aided

navigation through progress estimation. In *CVPR*, pages 6732–6740, 2019. 2

[28] Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. Compacter: Efficient low-rank hypercomplex adapter layers. In *NeurIPS*, pages 1022–1035, 2021. 1

[29] Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *ACL/IJCNLP*, pages 565–576, 2021. 1

[30] Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. Improving vision-and-language navigation with image-text pairs from the web. In *ECCV*, pages 259–274, 2020. 2

[31] Yuning Mao, Lambert Mathias, Rui Hou, Amjad Almahairi, Hao Ma, Jiawei Han, Scott Yih, and Madian Khabsa. Unipelt: A unified framework for parameter-efficient language model tuning. In *ACL*, pages 6253–6264, 2022. 1, 2, 3

[32] Yulei Niu, Hanwang Zhang, Zhiwu Lu, and Shih-Fu Chang. Variational context: Exploiting visual and textual context for grounding referring expressions. *IEEE TPAMI*, 43(1):347–359, 2021. 1

[33] Junting Pan, Ziyi Lin, Xiatian Zhu, Jing Shao, and Hongsheng Li. St-adapter: Parameter-efficient image-to-video transfer learning for action recognition. *CoRR*, abs/2206.13559, 2022. 1

[34] Yuankai Qi, Zizheng Pan, Yicong Hong, Ming-Hsuan Yang, Anton van den Hengel, and Qi Wu. The road to know-where: An object-and-room informed sequential bert for indoor vision-language navigation. In *ICCV*, pages 1655–1664, 2021. 2

[35] Yuankai Qi, Zizheng Pan, Shengping Zhang, Anton van den Hengel, and Qi Wu. Object-and-action aware model for visual language navigation. In *ECCV*, pages 303–317, 2020. 2

[36] Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. REVERIE: remote embodied visual referring expression in real indoor environments. In *CVPR*, pages 9979–9988, 2020. 2, 3, 6

[37] Yanyuan Qiao, Yuankai Qi, Yicong Hong, Zheng Yu, Peng Wang, and Qi Wu. HOP: history-and-order aware pre-training for vision-and-language navigation. In *CVPR*, 2022. 2

[38] Yanyuan Qiao, Yuankai Qi, Yicong Hong, Zheng Yu, Peng Wang, and Qi Wu. Hop+: History-enhanced and order-aware pre-training for vision-and-language navigation. *IEEE TPAMI*, 2023. 2

[39] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763, 2021. 1

[40] Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. LST: ladder side-tuning for parameter and memory efficient transfer learning. *CoRR*, abs/2206.06522, 2022. 1

[41] Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *CoRL*, pages 394–406, 2019. 2, 3, 6

[42] Hu Wang, Qi Wu, and Chunhua Shen. Soft expert reward learning for vision-and-language navigation. In *ECCV*, pages 126–141, 2020. 2

[43] Mohit Bansal Yi-Lin Sung, Jaemin Cho. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *CVPR*, pages 5217–5227, 2022. 1

[44] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C. Berg, and Tamara L. Berg. Modeling context in referring expressions. In *ECCV*, pages 69–85, 2016. 1

[45] Zhengkun Zhang, Wenya Guo, Xiaojun Meng, Yasheng Wang, Yadao Wang, Xin Jiang, Qun Liu, and Zhenglu Yang. Hyperpelt: Unified parameter-efficient language model tuning for both language and vision-and-language tasks. *arXiv preprint arXiv:2203.03878*, 2022. 1, 2, 3

[46] Fengda Zhu, Yi Zhu, Xiaojun Chang, and Xiaodan Liang. Vision-language navigation with self-supervised auxiliary reasoning tasks. In *CVPR*, pages 10009–10019, 2020. 2