Walk and Read Less: Improving the Efficiency of Vision-and-Language Navigation via Tuning-Free Multimodal Token Pruning

Anonymous ACL submission

Abstract

Large models have demonstrated state-of-the-art performance on Vision-and-Language Navigation tasks, but their high computational cost limits deployment in hardware-constrained environments. Token pruning reduces computation by decreasing the size of navigation inputs, offering a promising solution. However, in VLN tasks, input pruning can lead to information loss, causing the agent to take longer paths to determine when to stop, thus increasing computational demands and limiting efficiency gains. Moreover, attention-based pruning for instructions often fails to discard non-critical words, misspending valuable token budget. To improve navigation efficiency and address these challenges, we prune the navigation input from three angles. First, we divide the panoramic views into action and background tokens, preserving key information for action prediction while improving navigation efficiency by pruning the background views. Second, we prune nodes from the agent's navigation map to discourage backtracking and shorten paths. Finally, we leverage a Large Language Model to assess word importance in instructions, enabling us to accurately prune non-essential words. Experimental results show our methods significantly outperformed state-of-the-art pruning strategies in FLOPS efficiency, while maintaining higher accuracy across diverse VLN models and datasets.

1 Introduction

Vision-and-Language Navigation (VLN) enables AI agents to interpret natural language instructions and visual information to navigate effectively in their environments. While progress has been achieved in the *efficacy* of navigation agents (Fried et al., 2018; Tan et al., 2019; Hong et al., 2021; Chen et al., 2021, 2022; Wang et al., 2024c), advancements often come with computational costs too high for hardware-limited agents, underscoring the need to prioritize **navigation efficiency**, the task we are addressing here.

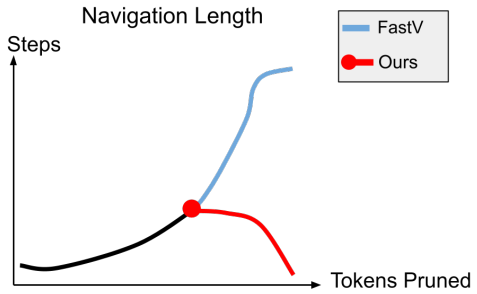

Figure 1: The proposed method reduces the navigation length through token pruning compared to other methods, in this example, FastV (Chen et al., 2025) on R2R (Anderson et al., 2018).

Token pruning enhances the computational efficiency of VLN models by reducing the size of their inputs, making it an attractive approach for lowering computational costs. Moreover, token pruning offers flexibility: the same VLN model can be adapted to meet different hardware constraints by simply adjusting the pruning rates, without the need for retraining. However, as more tokens are pruned, the navigation duration increases because the agent has less information to make correct and/or efficient decisions, as shown in Fig. 1. Longer paths incur higher computational costs, which undermines the benefits of token pruning. Furthermore, widely adopted attention-score-based text pruning methods fail to identify the important words in the instruction. As shown in Fig. 3, tokens such as "the" or "." are retained after pruning since they have high attention scores. Consequently, the VLN agent struggles to navigate effectively with instructions that lack critical information. To address these challenges we propose a "walk and read less" approach, where navigation is made efficient by targeting tokens in the input and addressing the above challenges.

We first observe that view inputs comprise of two token types: action and background tokens.

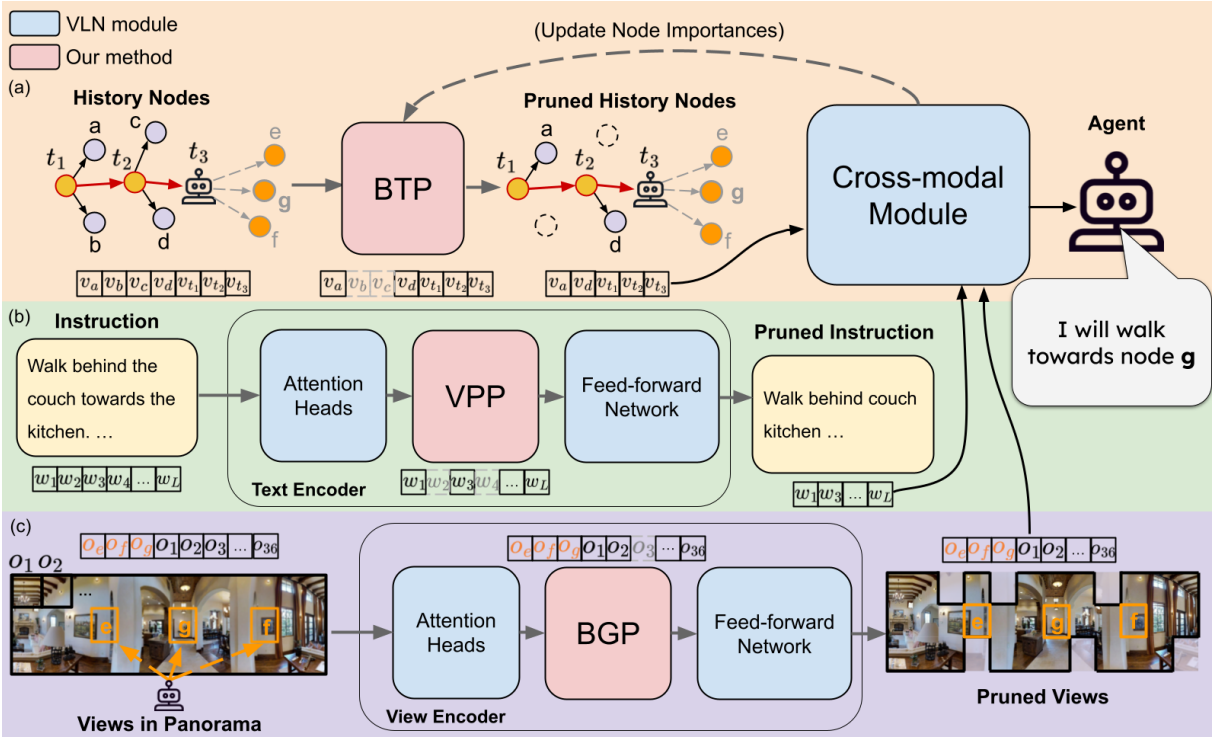

Figure 2: An overview of applying BGP, BTP, and VPP on a VLN model. (a) BTP prunes the unvisited history nodes from $\{v_a, v_b, ..., v_h\}$ on the navigation map constructed by the VLN models to discourage backtracking behaviors. (b) VPP prunes the irrelevant word tokens from the instruction tokens $\{w_1, w_2, ..., w_L\}$ to shorten textual inputs, and (c) BGP reduces the size of visual inputs by pruning the non-critical background views from $\{o_1, o_2, ..., o_{36}\}$.

While action tokens are essential for navigation, background tokens are ideal for pruning, which significantly reduces computational cost. We call this proposed process BackGround Pruning (BGP).

Background pruning lengthens navigation paths. We mitigate this with our BackTracking Pruning (BTP), which removes unvisited nodes from the navigation map of the environment to discourage backtracking and reduce costs (Fig. 1).

Attention-based pruning often fails to preserve key navigation words. To address this, we propose Vocabulary Priority Pruning (VPP), which retains essential words by building a "vocabulary of irrelevance." By prompting a large language model (e.g., Llama 3 (Dubey et al., 2024)) to identify non-critical words, VPP prunes these tokens before the rest. As shown in Fig. 3, critical words like "couch," "enter," and "doors" thus remain intact.

By combining BGP, BTP, and VPP (Fig. 2), the VLN agent navigates more efficiently by taking shorter paths ("walk less"), processing shorter instructions and fewer inputs of views ("read less"), while outperforming generic pruning strategies (i.e. prior work for VLM that have not been used for VLN yet). .

Our contributions are summarized as follows:

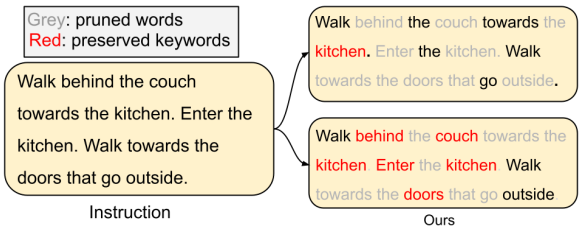

Figure 3: An example of words preserved in the instruction after pruning between prior works and ours.

1. We introduce BackGround view Pruning (BGP) to reduce input views and BackTracking Pruning (BTP) for more efficient pathing.
2. We introduce VPP, a strategy utilizes a Large Language Model (LLM) to prioritize the pruning of word tokens irrelevant to navigation, while preserving essential guidance tokens.
3. By integrating VPP, BTP, and BGP, our strategy achieves a performance that is almost twice as fast as the original model and 1.4 as fast as the model with the most recent pruning strategy in terms of GFLOPS (Giga FLoating-point OPerations per Second), with only a couple of percent points decrease in navigation success rate. These improvements are observed across multiple VLN models and datasets.

2 Related Work

VLN Datasets and Models. Multiple datasets (Anderson et al., 2018; Ku et al., 2020; Qi et al., 2020; Zhu et al., 2021; Hong et al., 2022) have been curated to address challenges that include long paths (Ku et al., 2020) and object localization (Qi et al., 2020; Zhu et al., 2021). To navigate through the environment, agents rely on transformer-based architectures for action prediction (Chen et al., 2021, 2022; An et al., 2023; Zhao et al., 2022; Huo et al., 2023; Wang et al., 2023a; Li and Bansal, 2023b; Wang et al., 2023b; Qiao et al., 2023a; Li and Bansal, 2023a; Wang et al., 2024b).

Advancements include history modules for navigation memory (Chen et al., 2021), backtracking mechanisms with topological mapping (Chen et al., 2022), enabling BTP as an efficiency-performance trade-off, and the use of synthesized visual data (Li and Bansal, 2023b). Wang et al. (2024b) further improved navigation accuracy with a debiasing strategy. Regarding VLN efficiency, Qiao et al. (2023b) explored Parameter-Efficient Transfer Learning, while Zhu et al. (2024) and Wang et al. (2024c) applied knowledge distillation to train smaller VLN models. These methods require new modules or model reconstruction, while our approach does not.

Token Reduction (Pruning and Merging). Recent works (Goyal et al., 2020; Wang et al., 2021; Kim et al., 2022; Liang et al., 2022; Bolya et al., 2022; Wei et al., 2023; Wang et al., 2024a; Zhang et al., 2024; Chen et al., 2025) primarily rely on attention scores to reduce sequence lengths in transformers. PoWER-BERT (Goyal et al., 2020) prunes input tokens based on self-attention. SpAtten (Wang et al., 2021) formally introduced cascade token and head pruning. Liang et al. (2022) addressed information loss by fusing pruned tokens, while Bolya et al. (2022) focused on removing feature redundancy. Chen et al. (2025) prune all unimportant tokens at a specific layer of the VLM. SparseVLM (Zhang et al., 2024) leverages external LLMs to assess visual token importance. Previous strategies assume attention scores reflect token importance, failing under high text pruning rates. They also ignore the impact of view token pruning on efficiency. We are the first to propose VLN-specific pruning to address these challenges.

3 Method: BGP, BTP, and VPP

Our method comprises three components: BackGround Pruning (BGP) for view pruning (Sec.3.2), BackTracking Pruning (BTP) for backtracking node pruning (Sec.3.3), and Vocabulary Priority Pruning (VPP) for instruction pruning (Sec. 3.4), which are integrated into VLN models (e.g., GOAT (Wang et al., 2024b)) as shown in Fig. 2.

3.1 The VLN Task, Inputs and Token Pruning

In VLN, an agent navigates based on an instruction of words $I = \{w_1, \ldots, w_L\}$. At each step, it processes views from a panorama $P = \{o_1, \ldots, o_N\}$ (Fried et al., 2018) and selects navigable locations $a \in A$ until it predicts a "STOP" action a_{stop}. Using the Matterport 3D simulator (Anderson et al., 2018; Ku et al., 2020; Qi et al., 2020), navigation is successful if the agent stops within 3 meters of the target; agent in REVERIE (Qi et al., 2020) must also locate the target object.

Most VLN models split the views in panoramas $P = \{o_1, ..., o_N\}$ into action views $O_{act} = \{o_1, ..., o_n\}$ and background views $O_{bgr} = \{o_{n+1}, ..., o_N\}$ providing visual context. The action views $\{o_1, ..., o_n\}$ correspond to navigable nodes $\{a_1, ..., a_n\}$ indicating navigable locations at the current step. In addition to views, DUET-based (Chen et al., 2022) models (Li and Bansal, 2023b; Wang et al., 2024b,c; Zhu et al., 2024) adopt a topological map as "history" to track the unvisited nodes $V_{unvisited} = \{v_1, v_2, ..., v_m\}$ and visited nodes $V_{visited} = \{v_{m+1}, v_{m+2}, ..., v_M\}$, with $V_{unvisited}$ enabling backtracking actions $\{a_1^{bt}, ..., a_m^{bt}\}$. The model's inputs are thus tri-modal, including I, P, and V. At each step, the model selects an action from stop, navigate, or backtrack using the policy $a = \pi(I, P, V)$, where $a \in \{a_{\text{stop}}, a_1, \ldots, a_n\} \cup \{a_1^{bt}, \ldots, a_m^{bt}\}$.

Given an instruction $I = \{w_1, w_2, ..., w_L\}$, views $P = \{o_1, ..., o_N\}$, and history nodes $V = \{v_1, v_2, ..., v_M\}$, token pruning reduces sequence lengths with a certain process, denoted as function f. We obtain the pruned sequences $I' = f_I(w_1, ..., w_L)$, $P' = f_P(o_1, ..., o_N)$, and $V' = f_V(v_1, ..., w_M)$, such that $|I'| < |I|$, $|P'| < |P|$, $|V'| < |V|$. The VLN model then selects an action $a = \pi(I', P', V')$ with lower computational cost. Efficient token pruning hinges on accurately assessing token importance, which we detail next.

3.2 Token Importance and Background Pruning

The token importance we adopted for BGP and BTP is attention-score-based, which is calculated during the transformer blocks in the VLN model.

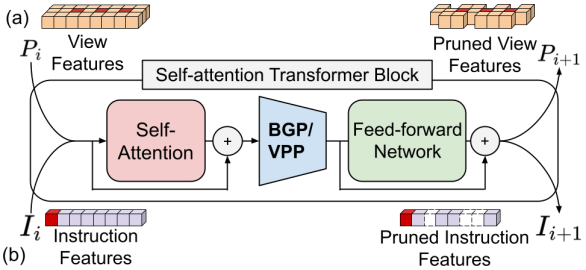

Figure 4: (a) BGP prunes view feature tokens based on their importance score before the residual connection "+". (b) VPP is similar to BGP, except a vocabulary of irrelevance is used to help decide token importance.

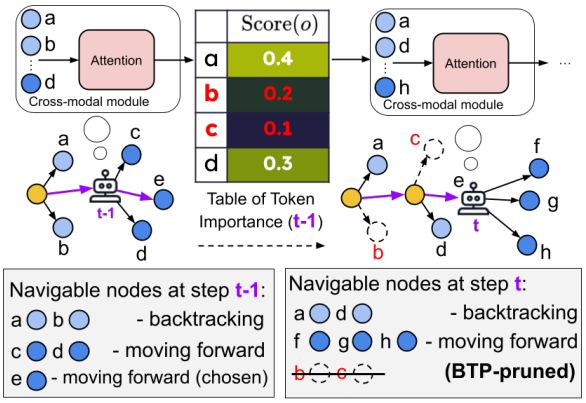

Figure 5: The BTP process. BTP prunes the unvisited nodes b and c at step t based on a table of token importance constructed from the cross-modal module at step $t-1$. Nodes b and c are no longer navigable at step t.

Concretely, during the navigation process, the panoramic views P are projected into a feature space using a pre-trained feature extractor, such as CLIP-B/16 (Radford et al., 2021), and enhanced with learned positional encodings that represent the viewing direction. This results in a sequence of N view tokens, each associated with a feature x. For simplicity, we denote the encoded view token sequence as P and each view feature token as o. Our token importance scores come from the transformer blocks processing P. Specifically, each transformer block consists of a self-attention module with attention heads, followed by a feed-forward network. At each head h in layer i, self-attention scores are computed to quantify the dependencies between tokens as follows:

$$\text{Attn}_{h_i} = \text{softmax}[Q_i \times K_i^T], \tag{1}$$

where $Q_i = P_i \times W_q^{h_i}$, $K_i = (P_i \times W_k^{h_i})$, and $W_q^{h_i}, W_k^{h_i}$ are trained weight matrices. The resulting Attn_{h_i} is a $N \times N$ matrix, where each entry $\text{Attn}_{h_i}[o_m, o_n]$ represents the normalized dependency of view o_n on view o_m. A higher value of $\text{Attn}_{h_i}[o_m, o_n]$ usually indicates that view o_n has a stronger influence on constructing the updated representation of o_m in the output P_{i+1}.

To determine the overall importance of each token, we aggregate the attention scores by summing each column of Attn_{h_i} across all attention heads H_i at layer i. This gives a vector of scores indicating the influence of each token o' on all tokens:

$$\text{Score}(o') = \sum_{h_i \in H_i} \sum_{o \in P} \text{Attn}_{h_i}[o, o']. \tag{2}$$

We interpret $\text{Score}(o')$ as the **importance score** of view o', which guides the pruning process by identifying less influential background views. Similar scores can be obtained for words w and history nodes v from the language module and cross-modal module, respectively. The module concludes by computing the latent feature $Z_i = \text{concatenate}(A_{h_i} \times (P_i \times W_v^{h_i}))$, where the concatenation is performed across all heads and $W_v^{h_i}$ is the weight matrix for each head.

BackGround view Pruning (BGP) reduces the size of O_{bgr} in P by removing k_{BGP} tokens at each layer of the vision transformer encoder. Inspired by Bolya et al. (2022), BGP is applied after computing the attention matrix Attn_{H_i} for P and before passing Z_i to the residual connection and feed-forward network (see Fig. 4). BGP retains all action view tokens O_{act} while pruning the k_{BGP} tokens with the lowest $\text{Score}(o)$ from O_{bgr} (i.e., removing their o and z values from O_{bgr} and Z_i). The pruned P_i and Z_i are then processed to yield P_{i+1}. Consequently, the final output P_{L+1} contains $k_{\text{BGP}} \cdot L$ fewer tokens than the original P, resulting in a smaller visual input for action prediction. Despite BGP reduces the visual inputs, it causes poorer action decisions, leading to longer navigation paths, which is addressed by BTP.

3.3 Backtracking Pruning

Backtracking in DUET-based VLN models (Chen et al., 2022) refers to returning to unvisited nodes from previous navigation steps. We propose Back-Tracking Pruning (BTP) to remove a subset of these unvisited nodes $V_{unvisited}$ from the history input V (nodes B & C in Fig. 5). Specifically, $V_{unvisited} = \{v_1, v_2, ..., v_m\}$ contains action view tokens from earlier steps that were not selected, i.e., $\{v_1, v_2, ..., v_m\} = \{o_1, o_2, ..., o_m\}$, where $\{o_1, o_2, ..., o_m\} \in O_{act}^{t'}$ at previous steps $t' < t$.

Method	R2R				RxR-English				REVERIE			
	Seen↑	Unseen↑	Steps↓	GFLOPS↓	Seen↑	Unseen↑	Steps↓	GFLOPS↓	Seen↑	Unseen↑	Steps↓	GFLOPS↓
Upper Bound, 100% Tokens												
VLN-GOAT	84.8	78.1	7.3	100	75.2	69.6	8.1	100	80.7	53.8	10.1	100
Retain ∼70% Tokens												
Random	79.1	**74.1**	7.0	78.1	73.3	68.1	8.4	84.2	71.5	48.1	11.5	82.3
FastV[1]	80.7	72.5	7.1	80.5	70.2	66.2	8.5	84.1	**80.2**	**54.2**	10.0	76.0
Cascade Pruning[2]	80.4	73.9	7.0	77.4	73.7	68.2	8.3	83.4	80.0	54.0	10.1	75.5
Cascade + ToMe[3]	80.1	73.8	6.7	74.6	73.3	67.7	8.3	83.6	71.6	47.9	11.4	81.3
Ours	**81.8**	73.2	**6.2**	**68.3**	**73.8**	68.0	8.3	**81.5**	79.3	52.4	10.0	**74.2**
Retain ∼60% Tokens												
Random	77.8	71.7	7.4	72.9	71.9	66.4	8.6	78.7	62.5	42.9	13.0	77.2
FastV[1]	77.8	69.3	7.6	77.2	68.2	63.4	8.7	79.9	63.2	42.7	13.3	81.5
Cascade Pruning[2]	77.9	71.3	7.5	74.1	72.1	66.8	8.6	78.3	63.3	43.0	13.4	79.0
Cascade + ToMe[3]	78.3	71.6	7.3	72.0	72.2	67.0	8.5	78.3	62.9	42.9	12.9	77.4
Ours	**81.6**	**72.9**	**6.3**	**61.8**	**73.4**	**68.2**	**8.4**	**75.5**	**78.8**	**52.1**	**10.1**	**65.9**
Retain ∼50% Tokens												
Random	73.8	67.3	8.3	68.6	69.4	64.2	9.0	74.0	54.0	35.0	14.4	70.3
FastV[1]	73.8	63.8	8.7	74.6	64.7	59.7	9.1	78.5	54.0	35.2	14.9	76.8
Cascade Pruning[2]	75.1	67.6	8.6	70.6	68.6	64.2	9.1	74.4	53.8	35.1	14.8	72.8
Cascade + ToMe[3]	76.4	67.5	7.9	69.5	70.3	65.7	8.6	75.4	53.8	35.4	14.6	71.7
Ours	**79.9**	**71.0**	**6.6**	**53.6**	**72.6**	**66.6**	8.6	**69.6**	**77.7**	**48.1**	**10.4**	**59.8**

Table 1: Performance of VLN-GOAT under various token pruning strategies. Metrics include navigation success rate ("Seen","Unseen"), average steps, and GFLOPS% relative to the base model. FastV[1](Chen et al., 2025) (textual & visual), Cascade pruning[2] (textual & visual) (Goyal et al., 2020; Wang et al., 2021), Cascade pruning and Token Merging (Bolya et al., 2022) (Cascade + ToMe[3], visual) are pruning techniques that are compared to our method.

Algorithm 1 Vocabulary Priority Pruning

procedure VPP($I, \mathcal{V}$, Scores, k_{vpp})
  $I_r \leftarrow$ empty sequence ▷ retained tokens
  $I_p \leftarrow$ empty sequence ▷ pruned tokens
  Sort w's in I by Score(w) from high to low
  for w in I **do**
    if w not in $\mathcal{V}$ **then**
      I_r.add(w)
    else
      I_p.add(w)
  if $k_{\text{vpp}} >$ length(I) - length(I_r) **then**
    for $i = 1$ to k_{vpp} **do**
      I_r.add(w_i)
  else
    for w in I_p **do**
      if length(I) - length(I_r) = k_{vpp} **then**
        break
      I_r.add(w)
  return I_r

Method	Retain Rate	Validation Seen			Validation Unseen		
		SR↑	SPL↑	GFLOPS↓	SR↑	SPL↑	GFLOPS↓
HAMT	-	70	67	100	63	58	100
HAMT (FastV)	0.7	67	64	80	**60**	**54**	82
	0.6	63	60	72	56	50	75
	0.5	58	55	**65**	50	45	68
HAMT (Ours$^-$)	0.7	**71**	**66**	**79**	59	53	**81**
	0.6	**69**	**65**	72	56	**51**	**74**
	0.5	**63**	**60**	66	**52**	**46**	68
DUET	-	79	73	100	72	60	100
DUET (FastV)	0.7	68	62	73	60	50	**73**
	0.6	62	56	66	54	44	**65**
	0.5	59	52	**56**	50	40	**55**
DUET (Ours)	0.7	**77**	**65**	**71**	**68**	**57**	75
	0.6	**75**	**69**	**64**	**66**	**55**	68
	0.5	**72**	**71**	57	**63**	**52**	59

Table 2: Model performance on the R2R dataset with different pruning strategies. Ours$^-$ indicates pruning without BTP.

To determine the most crucial nodes for successful navigation, we track the importance scores of these tokens from the attention heads of the last cross-modal transformer block at each step (e.g., 0.4 for node a in Fig. 5). At the beginning of the next step, BTP retains the top k_{BTP} unvisited nodes based on their latest Score(o) and discards the remainder, so that $V_{unvisited} = \{v_1, ..., v_{k_{\text{BTP}}}\}$.

BTP offers two main benefits: It reduces the size of the history input V to the cross-modal module, lowering computational cost, and it limits backtracking options from $\{a'_1, \ldots, a'_m\}$ to $\{a'_1, \ldots, a'_{k_{\text{BTP}}}\}$, enhancing path efficiency by reducing potential unnecessary backtracking.

3.4 Vocabulary Priority Pruning

VPP operates similarly to BGP by pruning instruction tokens $w \in I$ based on the importance Score(w) derived from the attention mechanism of the language transformer block (Fig. 4, Section 3.2). However, we found that the attention scores do not accurately reflect each word's importance for navigation. For example, as shown in Table 3, non-informative tokens, such as punctuation, often receive a high Score(w) and are retained, thereby wasting the token budget. To improve instruction

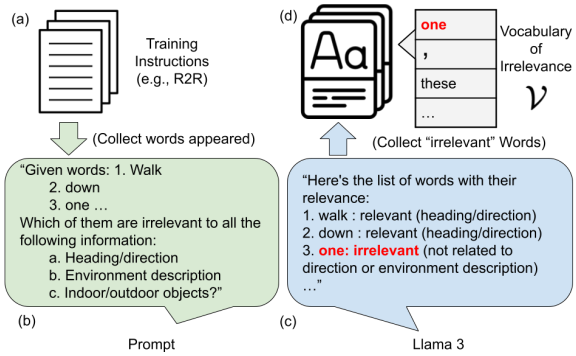

Figure 6: The construction of the "vocabulary of irrelavance" $\mathcal{V}$, it involves four steps, (a) collecting a lexicon of words appeared in the training VLN instructions, (b) building a prompt to ask the LLM to identify words irrelevant to navigation, (c) having the LLM generate classification results, and (d) collecting words classified as "irrelevant" into the vocabulary.

Retain %	Method	Instruction Tokens (grey ones are pruned)
100	-	<s>Exit the room . Turn right . Start down the stairs and stop 3 steps down . </s>
50	VPP	<s>Exit the room . Turn right . Start down the stairs and stop 3 steps down . </s>
50	Att. Scores	<s>Exit the room . Turn right . Start down the stairs and stop 3 steps down . </s>
25	VPP	<s>Exit the room . Turn right . Start down the stairs and stop 3 steps down . </s>
25	Att. Scores	<s>Exit the room . Turn right . Start down the stairs and stop 3 steps down . </s>

Table 3: Instruction tokens retained under different pruning rates. VPP is more effective in preserving key information than attention-score pruning, which fails to keep navigation-relevant words.

pruning, we prompt an LLM to construct a "vocabulary of irrelevance" $\mathcal{V}$ that identifies unimportant words based on its general knowledge rather than attention scores. The construction process is illustrated in Fig. 6. We tokenized the training data to form a lexicon and prompted Llama 3 (Dubey et al., 2024) to label each word as "relevant" or "irrelevant" for navigation based on its association with direction/heading, environment description, or indoor/outdoor objects. Words labeled "irrelevant" were compiled into $\mathcal{V}$; see Appendix A.1 for an example prompt and vocabulary.

At each layer of the language transformer encoder, VPP prunes k_{vpp} instruction tokens via a two-step process. First, it filters out all words present in $\mathcal{V}$. If the number of filtered tokens is less than k_{vpp}, the remaining tokens are pruned based on their attention scores $\text{Score}(w)$ until the requirement is met. Otherwise, if more tokens are filtered than necessary, some are reinstated according to $\text{Score}(w)$ to preserve additional information. The VPP process is detailed in Algorithm 1.

VPP more effectively retains essential words in instructions, enabling the navigation system to function normally even at higher pruning rates compared to attention-based strategies (Table 3).

By combining BGP, BTP, and VPP, we obtain the pruned views P_{BGP}, history nodes V_{BTP}, and instruction I_{VPP} where the lengths are $L \cdot k_{BGP}$, $M - k_{\text{BTP}}$, and k_{VPP} shorter than P, V, I. The VLN model then calculates the prediction action a by $\pi(P_{\text{BGP}}, V_{\text{BTP}}, I_{\text{VPP}})$, resulting in lower computational cost, as detailed in the next section.

4 Experiments

Datasets. We evaluated our proposed pruning strategies on three VLN benchmarks: R2R (Anderson et al., 2018), RxR (Ku et al., 2020), and REVERIE (Qi et al., 2020). For each dataset, token pruning was tested on two splits, the validation "seen" split and validation "unseen" split.

Evaluation Metrics. Our primary *efficacy metric* is navigation Success Rate (SR), which measures the agent's ability to navigate under pruning conditions. Additional efficacy metrics are reported in our result tables.

Our *efficiency metric* is a simplified version of a GFLOPS-based formula introduced by Wang et al. (2024c):

$$G_{\text{total}} = G_{lan}(I) + D \cdot (G_{vis}(P) + G_{cm}(I, P, V)) + c,$$

where D is the number of the decision steps ("steps"), G_{lan} is the GFLOPS of language module getting instruction features, G_{vis} is the GFLOPS of visual module processing view features, G_{cm} is the GFLOPS of the cross-modal module predicting actions from all input features, and c is the cost from other sources, if any. We assess efficiency improvement in terms of GFLOPS, as detailed in Section 4. To convey efficiency improvements, we report the ratio of GFLOPS after pruning to GFLOPS before pruning (in %):

$$\text{GFLOPS} = G_{\text{pruned}} / G_{\text{original}}. \quad (3)$$

Our computation cost is computed using the Python toolkit `thop`.

VLN Models. We evaluated our token pruning strategies on three VLN models: HAMT (Chen et al., 2021), DUET (Chen et al., 2022), and GOAT (Wang et al., 2024b). All experiments use

Method	Retain Rate	Validation Unseen				
		SR↑	SPL↑	RGS↑	RGSPL↑	GFLOPS↓
HAMT	-	32	29	19	17	100
HAMT (FastV)	0.7	**32**	29	18	16	73
	0.6	30	27	16	14	68
	0.5	27	24	14	12	65
HAMT (Ours$^-$)	0.7	31	29	**19**	**17**	**69**
	0.6	**31**	**28**	**18**	**17**	**64**
	0.5	**29**	**26**	**17**	**15**	**59**
DUET	-	47	34	32	23	100
DUET (FastV)	0.7	39	28	23	16	82
	0.6	34	24	19	13	73
	0.5	30	20	16	11	64
DUET (Ours)	0.7	**47**	**33**	**31**	**22**	**80**
	0.6	**45**	**32**	**31**	**22**	**70**
	0.5	**43**	**30**	**28**	**20**	**62**

Table 4: Performance of pruning strategies on REVERIE data Ours$^-$ indicates pruning without BTP. Efficacy is measured in Success Rate (SR) Per Length (SPL), Remote Grounding Success rate (RGS), and Remote Grounding Success rate Per Length (RGSPL).

BGP	BTP	VPP	SR↑	SPL↑	Steps↓	GFLOPS↓
✗	✗	✗	78.1	67.6	7.0	100
✓	✗	✗	75.4	62.4	8.0	72.3
✗	✓	✗	74.6	66.0	6.6	90.1
✗	✗	✓	75.5	64.6	7.3	95.8
✓	✓	✗	72.2	61.9	7.2	60.7
✓	✗	✓	73.9	60.7	8.2	65.7
✗	✓	✓	72.4	63.4	6.81	85.5
✓	✓	✓	71.0	60.3	7.3	54.4

Table 5: Ablation of BGP, BTP, and VPP contribution to efficiency and influence to SR and SPL on R2R validation unseen split.

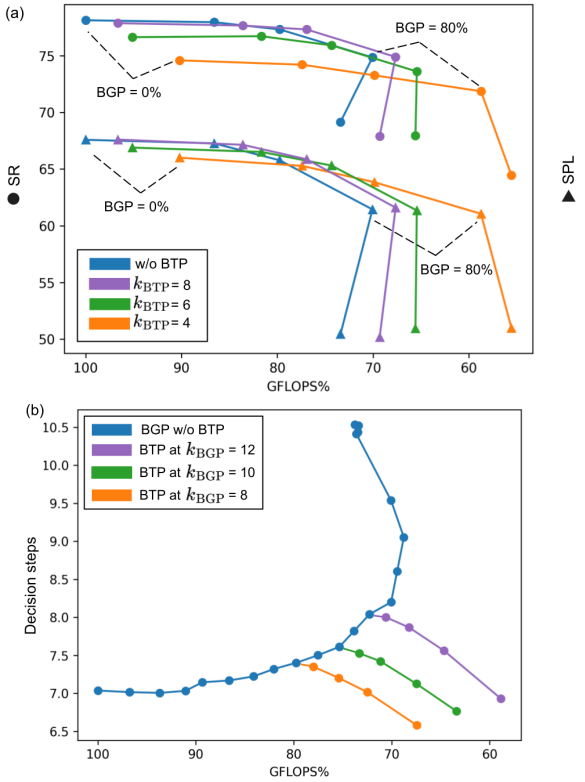

Figure 7: (a) Success rates and GFLOPS of different pruning rates for BGP, and BGP with various BTP settings. (b) Average Steps for navigation and GFLOPS of different pruning rates for BGP with no BTP, and BTP with various fixed BGP settings.

the pre-tuned parameters provided by the respective authors. We extensively tested VLN-GOAT for its outstanding performance across datasets. VLN-GOAT comprises six transformer layers in the language module, two layers in the view module, and three layers in the cross-modal module.

Pruning Strategies. We are the first to apply multi-modal token pruning to the VLN task. For baseline comparisons, we selected several pruning strategies based on VLN model performance and if they are applicable to VLN inputs. For instruction pruning, we compared our method against three approaches, removing tokens randomly, cascade pruning (Goyal et al., 2020; Wang et al., 2021), which prunes a portion of tokens at each transformer layer, and FastV (Chen et al., 2025), which prunes the required number of tokens all at once at a specific layer. For view input pruning, we only consider background pruning without removing any action views, as the SR drops significantly even with minimal action view removal (see Appendix A.3). We also evaluated merging tokens with similar key vector features (Bolya et al., 2022). For the settings of BGP, BTP, VPP, we retained the same amount of views and words as the baselines, while always keeping at most 8 unvisited node for BTP.

4.1 Comparison with Baselines

The performance of the base model GOAT under various token pruning strategies and rates across three datasets is reported in Table 1. Our approach consistently yields greater GFLOPS reduction and shorter navigation paths while maintaining comparable or higher Success Rates (SR) than baselines. At a 50% token pruning rate, our VLN model attains SRs of 71.0%, 66.6%, and 48.1% on the Unseen splits of R2R, RxR-English, and REVERIE, compared to 67.6%, 65.7%, and 35.4% from baselines (last row of Table 1). Additionally, our method saves up to 15 percent points (pp) in GFLOPS and reduces navigation steps, demonstrating enhanced pathing efficiency.

We validated our method on HAMT (Chen et al., 2021) and DUET (Chen et al., 2022) using the R2R

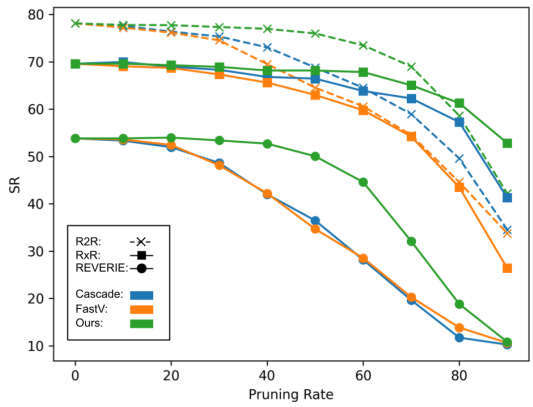

Figure 8: Success rates for different pruning strategies and rates for BGP and BVP.

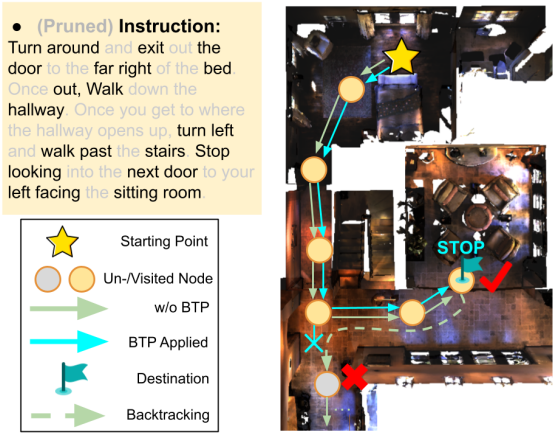

Figure 9: An example of navigation with our strategies applied. BTP prevents a long and failed navigation path by pruning unvisited nodes.

and REVERIE datasets, compared to FastV (Chen et al., 2025) (see Tables 2 and 4). In DUET, our method outperforms FastV by 5pp to 10pp in SR, SPL, RGS, and RGSPL with similar GFLOPS. In HAMT, where BTP cannot be applied, our approach still shows superior SR and SPL on R2R and outperforms FastV in both navigation performance and efficiency on REVERIE.

To illustrate the impact of our pruning strategy, Fig. 9 shows a navigation example. VPP preserves key instruction words, enabling the agent to follow and complete the route. Meanwhile, BTP "forces" the agent to stop at the correct destination by removing potential backtracking nodes; without it, unpruned unvisited nodes would cause backtracking, resulting in longer paths and navigation failure.

4.2 Ablation Study

We conducted an ablation study to assess the effectiveness of BTP, BGP, and VPP under a 50% pruning rate setting (Table 5). BGP yields the highest GFLOPS reduction (27.7pp), indicating that background views are highly prunable. BTP adds roughly 10pp in GFLOPS reduction and reduces average navigation steps by one, while VPP improves GFLOPS by 5pp, due to R2R's short instructions. VPP benefits the model more if the dataset contains longer instructions (shown in Appendix A.4).

To further examine BTP's impact, we evaluated various BTP settings in combination with BGP at different pruning rates on the R2R dataset, tracking changes in SR, SPL, and navigation steps (Fig. 7). In part (a), retaining 6 or 8 unvisited nodes per step caused minimal drops in SR and SPL while further reducing GFLOPS. However, a more aggressive setting—retaining only 4 unvisited nodes (orange)—resulted in a 3–4% SR loss at lower BGP rates, though efficiency improved significantly (over 10pp GFLOPS reduction). Overall, moderate BTP provides additional efficiency gains with minimal impact on success rate, whereas more aggressive BTP clearly trades success rate for efficiency. When combined with other background pruning strategies and applied to various datasets, BTP further enhances navigation efficiency with minimal SR loss (Appendix A.5). Fig. 7(b) shows that BTP reduces navigation steps across various BGP settings, further supporting our findings.

Lastly, we compare VPP with Cascade Pruning and FastV for instruction token removal on R2R, RxR, and REVERIE (Fig. 8). VPP (green) consistently outperforms the other two strategies across most pruning rates, with optimal improvements in the 40–60% range. Our experiments show that the same vocabulary can be effectively applied across different text tokenizers and datasets without the need for reconstruction (Appendix A.6).

5 Conclusion

Our work introduces three token pruning strategies: BGP, BTP, and VPP for VLN that effectively reduce input size while addressing issues of increased navigation path length and inadequate retention of critical instruction information observed with generic pruning methods. Experiments on the R2R, RxR, and REVERIE datasets show that our approach outperforms them in both navigation success rates and computational efficiency. Overall, our robust, model- and dataset-agnostic solution significantly lowers computational costs without sacrificing performance, paving the way for more efficient deployment of VLN models in resource-constrained environments.

6 Limitations

A majority of generic token pruning strategies, including our BGP and BTP, assume that attention scores accurately reflect the true importance of token features for navigation success. However, this assumption is inherently flawed. While VPP partly mitigates this issue for instruction pruning, an analogous solution for view pruning would incur expensive costs, such as those associated with gradient- or perturbation-based saliency. Moreover, as shown in Table 1, random token pruning performs nearly as well as attention-based methods—except for our approach. This suggests that many VLN tokens may contribute similarly to navigation, yielding comparable efficiency gains and success rate losses regardless of the pruning method. We hope our work serves as a starting point for understanding token importance in VLN for both text and visuals, and inspires further development of effective, cost-efficient indicators for VLN token pruning.

References

Dong An, Yuankai Qi, Yangguang Li, Yan Huang, Liang Wang, Tieniu Tan, and Jing Shao. 2023. BEVBERT: Multimodal Map Pre-training for Language-guided Navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2737–2748.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. 2018. 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.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. 2022. Token Merging: Your ViT But Faster. *arXiv preprint arXiv:2210.09461*.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. 2025. An Image is Worth 1/2 Tokens After Layer 2: Plug-and-play Inference Acceleration for Large Vision-language Models. In *European Conference on Computer Vision*, pages 19–35. Springer.

Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. 2021. History Aware Multimodal Transformer for Vision-and-language Navigation. *Advances in neural information processing systems*, 34:5834–5847.

Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. 2022. Think Global, Act Local: Dual-scale Graph Transformer for Vision-and-language Navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16537–16547.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*.

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. *Advances in neural information processing systems*, 31.

Saurabh Goyal, Anamitra Roy Choudhury, Saurabh Raje, Venkatesan Chakaravarthy, Yogish Sabharwal, and Ashish Verma. 2020. Power-BERT: Accelerating BERT Inference via Progressive Word-vector Elimination. In *International Conference on Machine Learning*, pages 3690–3699. PMLR.

Yicong Hong, Zun Wang, Qi Wu, and Stephen Gould. 2022. Bridging the Gap between Learning in Discrete and Continuous Environments for Vision-and-language Navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15439–15449.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2021. VLN BERT: A Recurrent Vision-and-language BERT for Navigation. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 1643–1653.

Jingyang Huo, Qiang Sun, Boyan Jiang, Haitao Lin, and Yanwei Fu. 2023. GeoVLN: Learning Geometry-enhanced Visual Representation with Slot Attention for Vision-and-language Navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23212–23221.

Sehoon Kim, Sheng Shen, David Thorsley, Amir Gholami, Woosuk Kwon, Joseph Hassoun, and Kurt Keutzer. 2022. Learned Token Pruning for Transformers. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 784–794.

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*.

Jialu Li and Mohit Bansal. 2023a. Improving vision-and-language navigation by generating future-view image semantics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10803–10812.

Jialu Li and Mohit Bansal. 2023b. PanoGen: Text-conditioned Panoramic Environment Generation for Vision-and-language Navigation. *Advances in Neural Information Processing Systems*, 36:21878–21894.

Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. 2022. Not all patches are what you need: Expediting vision transformers via token reorganizations. *arXiv preprint arXiv:2202.07800*.

Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. 2020. REVERIE: Remote Embodied Visual Referring Expression in Real Indoor Environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9982–9991.

Yanyuan Qiao, Yuankai Qi, Yicong Hong, Zheng Yu, Peng Wang, and Qi Wu. 2023a. Hop+: History-enhanced and order-aware pre-training for vision-and-language navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(7):8524–8537.

Yanyuan Qiao, Zheng Yu, and Qi Wu. 2023b. VLN-PETL: Parameter-Efficient Transfer Learning for Vision-and-Language Navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15443–15452.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning Transferable Visual Models from Natural Language Supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Hao Tan, Licheng Yu, and Mohit Bansal. 2019. Learning to Navigate Unseen Environments: Back Translation with Environmental Dropout. *arXiv preprint arXiv:1904.04195*.

Hanrui Wang, Zhekai Zhang, and Song Han. 2021. Spatten: Efficient sparse attention architecture with cascade token and head pruning. In *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 97–110. IEEE.

Hongjie Wang, Bhishma Dedhia, and Niraj K Jha. 2024a. Zero-TPrune: Zero-shot Token Pruning Through Leveraging of the Attention Graph in Pre-trained Transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16070–16079.

Liuyi Wang, Zongtao He, Ronghao Dang, Mengjiao Shen, Chengju Liu, and Qijun Chen. 2024b. Vision-and-Language Navigation via Causal Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13139–13150.

Liuyi Wang, Zongtao He, Mengjiao Shen, Jingwei Yang, Chengju Liu, and Qijun Chen. 2024c. Magic: Meta-ability Guided Interactive Chain-of-distillation for Effective-and-efficient Vision-and-language Navigation. *arXiv preprint arXiv:2406.17960*.

Liuyi Wang, Zongtao He, Jiagui Tang, Ronghao Dang, Naijia Wang, Chengju Liu, and Qijun Chen. 2023a. A Dual Semantic-aware Recurrent Global-adaptive Network for Vision-and-Language Navigation. *arXiv preprint arXiv:2305.03602*.

Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, and Shuqiang Jiang. 2023b. GridMM: Grid Memory Map for Vision-and-language Navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15625–15636.

Siyuan Wei, Tianzhu Ye, Shen Zhang, Yao Tang, and Jiajun Liang. 2023. Joint Token Pruning and Squeezing towards More Aggressive Compression of Vision Transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2092–2101.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, et al. 2024. SparseVLM: Visual token sparsification for efficient vision-language model inference. *arXiv preprint arXiv:2410.04417*.

Yusheng Zhao, Jinyu Chen, Chen Gao, Wenguan Wang, Lirong Yang, Haibing Ren, Huaxia Xia, and Si Liu. 2022. Target-driven Structured Transformer Planner for Vision-language Navigation. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4194–4203.

Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. 2021. Soon: Scenario Oriented Object Navigation with Graph-based Exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12689–12699.

Junyou Zhu, Yanyuan Qiao, Siqi Zhang, Xingjian He, Qi Wu, and Jing Liu. 2024. MiniVLN: Efficient Vision-and-Language Navigation by Progressive Knowledge Distillation. *arXiv preprint arXiv:2409.18800*.

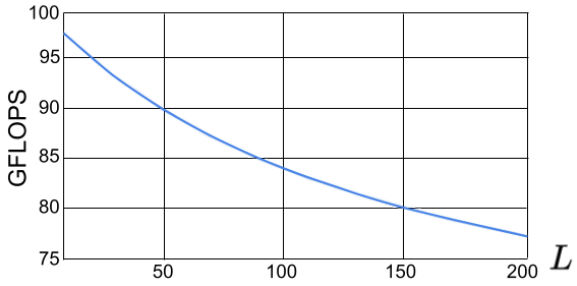

Figure 10: GFLOPS curve over different instruction lengths with 50% token pruning rate.

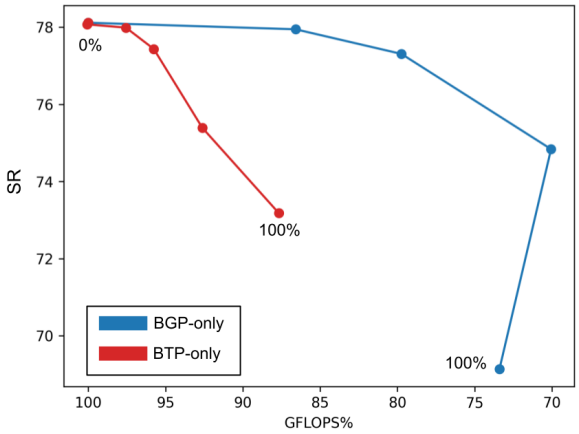

Figure 11: A comparison of applying BGP and BTP alone.

A Appendix

A.1 Vocabulary Example

We provide an example of how we prompt the LLM to construct the vocabulary of irrelevance, with explanations from Llama 3 why such tokens are classified to be relevant or irrelevant (see Table 6).

A.2 VLN Datasets Statistics

The statistic of VLN dataset statistic is given in Table 7. The tokens numbers are averaged over steps of all navigation tasks.

A.3 Pruning Action Views

In VLN, action views are essential as they provide directional cues and define navigable paths. As shown in Table 8, pruning action views drastically reduces SR—resulting in longer paths or halted movement—whereas BGP maintains performance with smoother path lengths, making it ideal for improving navigation efficiency.

A.4 Instruction Length Influence on Efficiency Improvement

VPP's efficiency gains are closely related to instruction length. Under the default settings for R2R, RxR, and REVERIE (36 views, 14 history nodes, and a 7-step navigation), we evaluated the efficiency improvement from a 50% instruction pruning rate across different instruction lengths (see Fig. 10). The results show that a 50-token instruction achieves a 10 percentage point GFLOPS reduction, which increases to 20 percentage points for a 150-token instruction.

A.5 BTP alone, and combined with different view pruning strategies

We compare the effectiveness of BackTracking Pruning (BTP) with Background Pruning (BGP) in Figure 11. Our results indicate that pruning background views is far more effective than pruning backtracking nodes, making BTP alone impractical. Thus, combining BTP with background view pruning is more preferable to counteract the increased path length resulting from view pruning.

Specifically, we evaluated the GFLOPS improvements when applying BTP with cascade token pruning, FastV, and Token Merging (ToMe) across the R2R, RxR, and REVERIE datasets (see Fig. 12). Our findings show that incorporating BTP reduces GFLOPS by approximately 5 percentage points at high BGP rates (e.g., 80%), while resulting in a Success Rate drop of less than 0.5 percentage points.

A.6 Adopting a Single Vocabulary of Irrelevance to Different Language Models and Datasets

We observed that the vocabulary constructed by the LLM can be applied across different language modules and datasets without needing to re-build it for each case. To validate this, we built vocabularies from different datasets and from the tokenizers of three pre-trained language models (BERT, RoBERTa, XLM-RoBERTa), then applied them to a VLN language module initialized with weights from a different language model (Table 9). The improvements from VPP were similar regardless of the vocabulary source, demonstrating that VPP is a robust, model- and dataset-agnostic pruning strategy.

Prompt	Given the following set of words: “walk”, “down”, ..., “landing”, can you point out which of them are irrelevant to the following types of information: 1. A direction to go; 2. Describing the environment; 3. Object(s) in the indoor/outdoor environments. Please don’t change the word in the quotation mark and explain why. Please answer in the following: format: “{word} : relevant/irrelevant {explanation}”	
Word	Relevance	Explanation
walk	relevant	direction to go
down	relevant	direction to go
one	irrelevant	not related to direction or environment description
flight	irrelevant	not related to indoor environment navigation
of	irrelevant	preposition, not related to direction or environment description
stairs	relevant	object in the environment
and	irrelevant	conjunction, not related to direction or environment description
stop	relevant	direction to go
the	irrelevant	article, not related to direction or environment description
landing	relevant	object in the environment

Table 6: Our prompt to Llama 3, and examples of words in the vocabulary of irrelevance, explained by the LLM why they are relevant/irrelevant.

Dataset	R2R		RxR-english		REVERIE	
	Seen	Unseen	Seen	Unseen	Seen	Unseen
Size	1020	2349	2939	4551	1423	3521
History nodes size	14		19		16	
View tokens	38		38		42	
Instruction tokens	32		127		20	

Table 7: Statistics of R2R, RxR-English, and REVERIE regarding token pruning. The numbers are averaged over all navigation tasks and steps.

Pruning Rate	All Views			BGP		
	SR↑	Steps↓	GFLOPS↓	SR↑	Steps↓	GFLOPS↓
0	78	7	100	78	7	100
0.25	62	9	108	78	7	87
0.4	44	11	111	77	7	80
0.7	13	6	55	75	8	70
0.9	1	1	14	69	11	73

Table 8: A comparison of pruning view tokens including action views and BGP. Pruning a large fraction of action views caused significant SR and GFLOPS drops while pruning at a lower rate led to higher computation cost.

Target \ Source	R2R	RxR	REVERIE
R2R (68.8)	76.0 (+7.2)	75.5 (+6.7)	74.0 (+5.2)
RxR (66.4)	68.0 (+1.6)	68.2 (+1.8)	68.2 (+1.8)
REVERIE (36.5)	49.8 (+13.3)	47.9 (+11.4)	50.0 (+13.5)
Target \ Source	**RoBERTa**	**BERT**	**XLM-R**
RoBERTa (66.4)	68.2 (+1.8)	68.4 (+2.0)	69.0 (+2.6)
BERT (59.3)	65.4 (+6.1)	66.0 (+6.7)	65.7 (+6.4)
XLM-R (40.4)	55.4 (+15.0)	56.1 (+15.7)	55.4 (+15.0)

Table 9: Navigation success rates using a vocabulary of irrelevance built on the source dataset (top) or tokenizer (bottom) and applied during inference on the target dataset/tokenizer. The leftmost column (e.g., 68.8) shows success rates with a 50% instruction pruning rate.

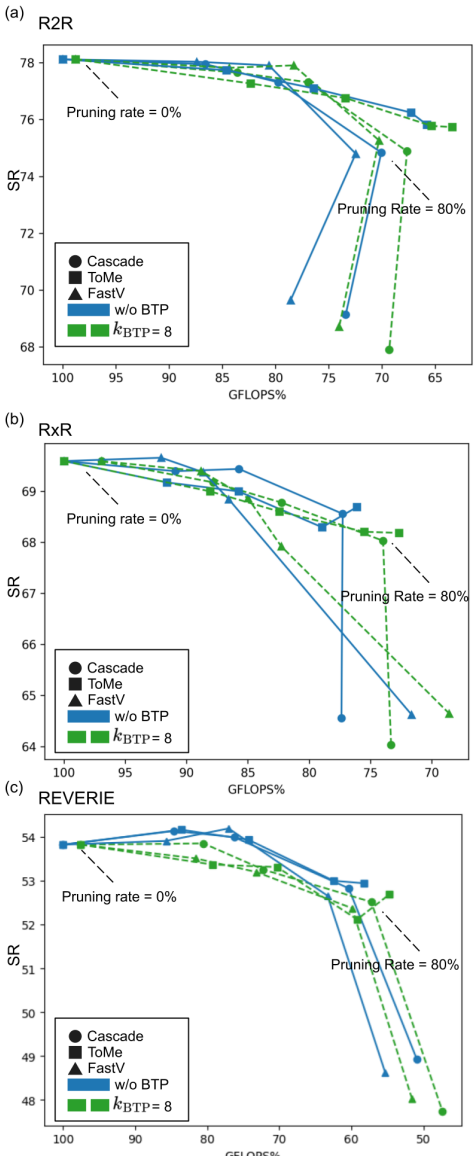

Figure 12: SR and GFLOPS curves for different view pruning rates with and without BTP. Performance is evaluated on the R2R, RxR and REVERIE datasets.