CS585: Multimodal Learning Topics for Vision-and-Language Navigation

Wenda Qin

4/18/2024

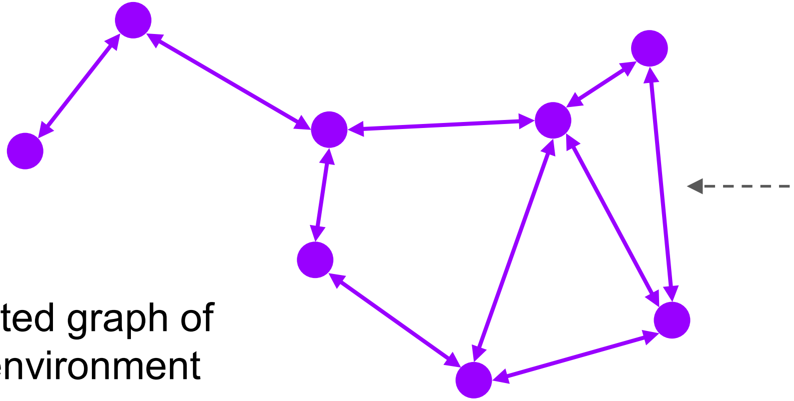
Table of Content:

1. Introduction to Vision-and-Language Navigation (VLN)
2. Attention mechanism: Attention between vision and language features.
3. Data augmentation: From instruction generation to scene generation

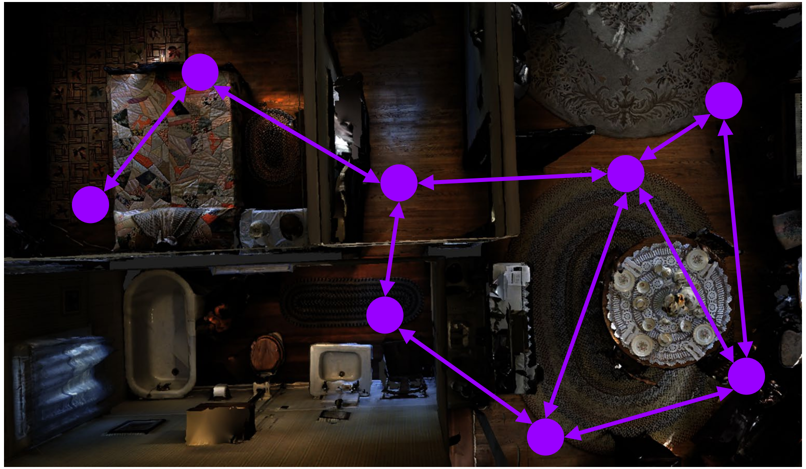
Definition of the VLN Problem:

Goal: Build a AI system that guides robot with camera (vision) traveling from A to B, given an instruction by human (language).

Rule 1: Navigation happens in simulated environment, represented by a connected graph. The nodes of the graph are locations (viewpoints) to which the agent can move during the navigation. The edges between nodes indicate whether the robot can move between nodes or not.



Connected graph of scene/environment

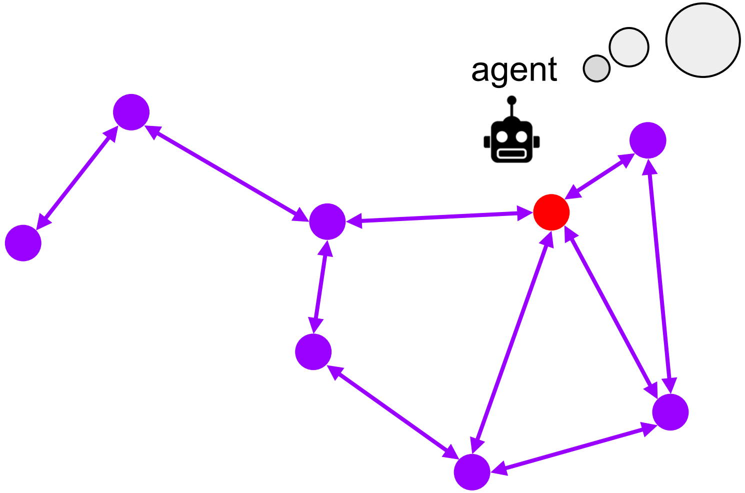


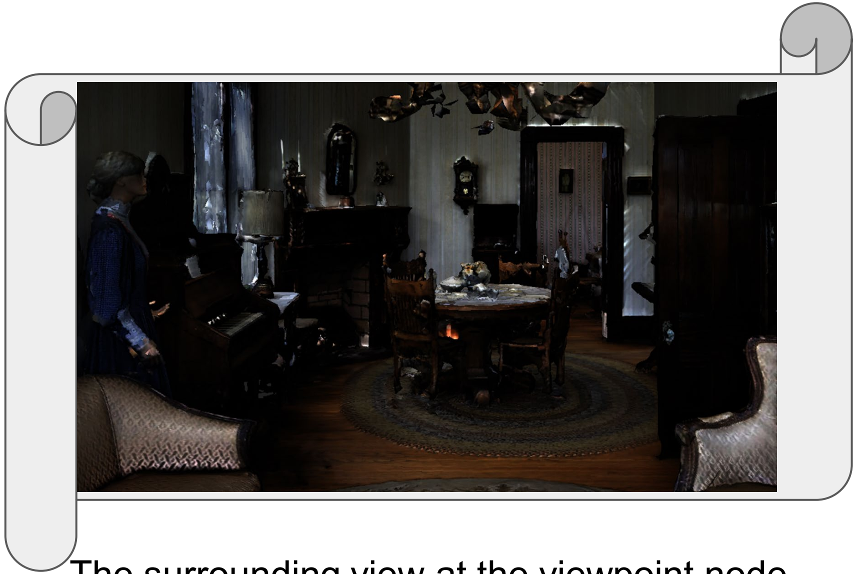
Definition of the VLN Problem:

Rule 2: To navigate through the scene, the **agent** (the robot in the simulated scene) is given two types of information:

1. An **instruction** that tells the agent where to go and stop at the end. The instruction won't change during the navigation.
2. The surrounding **view** of the node the agent currently stands in. We call these nodes "**viewpoints**."

"Walk into the room, go inside another room, stop on the other side of the bed."



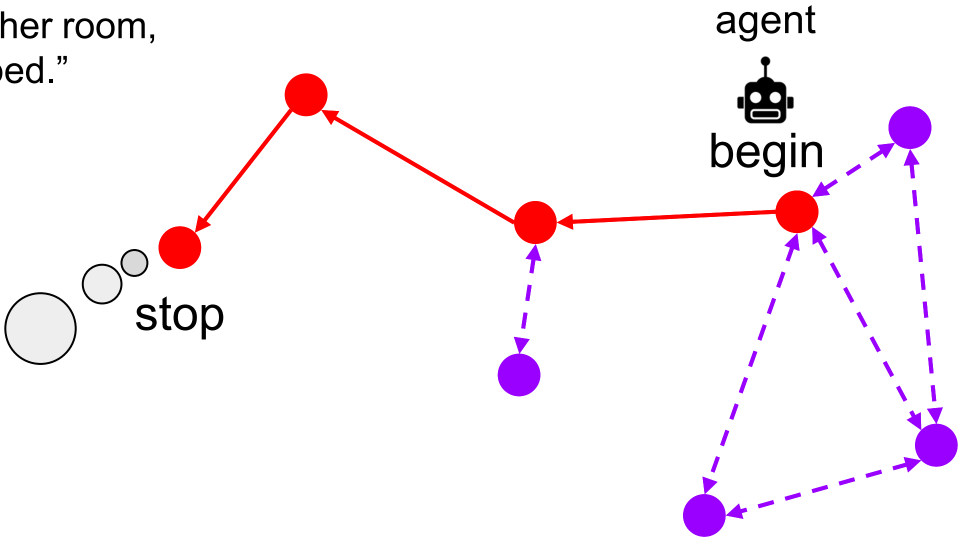


The surrounding view at the viewpoint node

Definition of the VLN Problem:

Rule 3: The agent needs to decide where to go next from a set of “navigable” viewpoints (nodes are “navigable” when nodes are connected by an edge) or stop. The navigation ends when the agent decides to stop. The navigation is considered successful if the destination is close enough to the ground truth destination.

“Walk into the room, go inside another room, stop on the other side of the bed.”



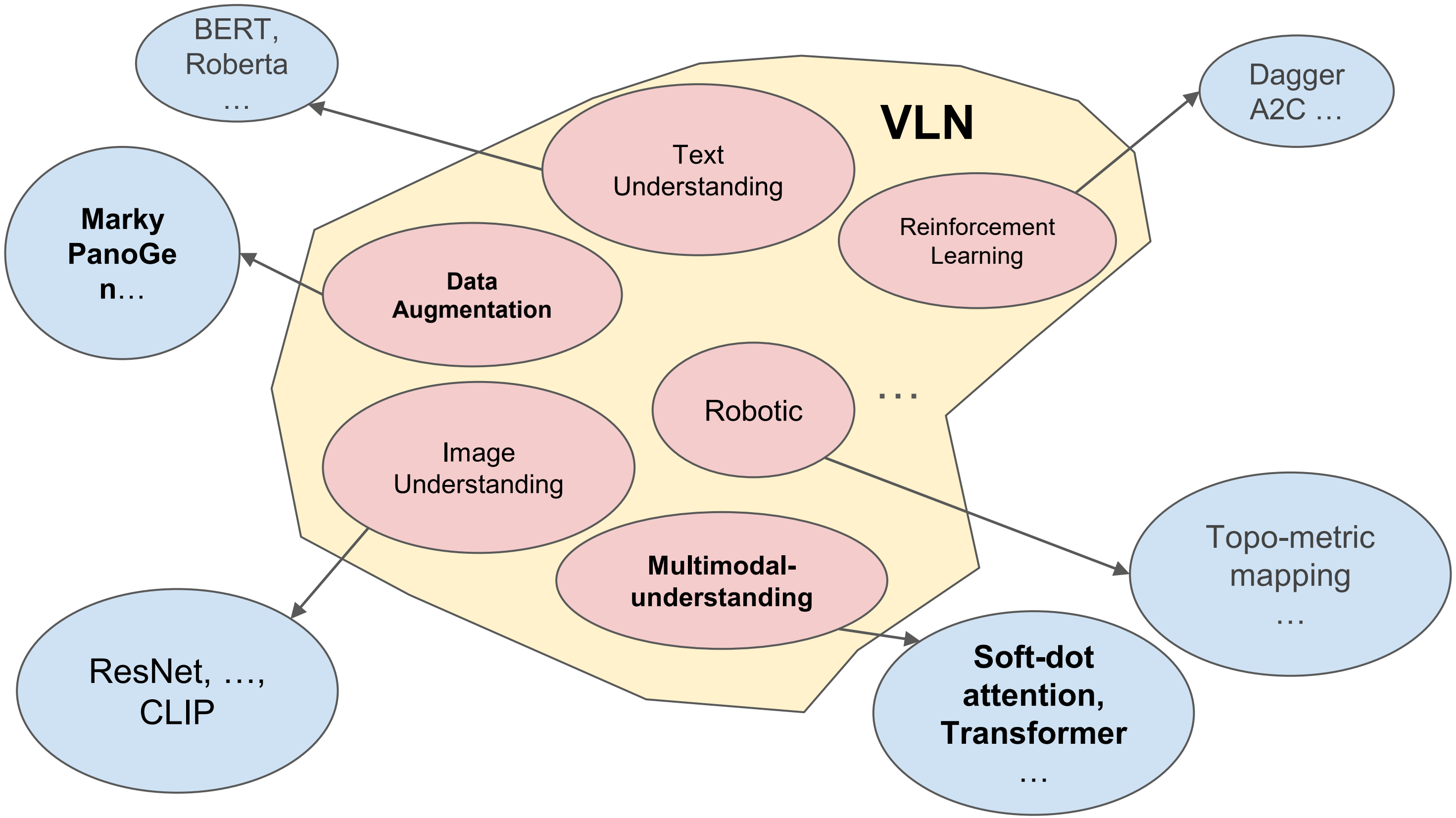
Definition of the VLN Problem:

Methods	Val Unseen				Test Unseen			
	NE↓	OSR↑	SR↑	SPL↑	NE↓	OSR↑	SR↑	SPL↑
Seq2Seq [1]	7.81	28	21	-	7.85	27	20	-
SF [32]	6.62	45	36	-	6.62	-	35	28
Chasing [62]	7.20	44	35	31	7.83	42	33	30
RCM [38]	6.09	50	43	-	6.12	50	43	38
SM [33]	5.52	56	45	32	5.67	59	48	35
EnvDrop [39]	5.22	-	52	48	5.23	59	51	47
AuxRN [71]	5.28	62	55	50	5.15	62	55	51
NvEM [36]	4.27	-	60	55	4.37	66	58	54
SSM [19]	4.32	73	62	45	4.57	70	61	46
PREVAL [10]†	4.71	-	58	53	5.30	61	54	51
AirBert [12]†	4.10	-	62	56	4.13	-	62	57
RecBert [48]†	3.93	-	63	57	4.09	70	63	57
REM [40]	3.89	-	64	58	3.87	72	65	59
HAMT [22]†	3.65	-	66	61	3.93	72	65	60
HOP+ [72]†	3.49	-	67	61	3.71	-	66	60
EnvEdit* [42]†	3.24	-	69	64	3.59	-	68	64
TD-STP [49]†	3.22	76	70	63	3.73	72	67	61
DUET [24]†	3.31	81	72	60	3.65	76	69	59
BEVBert (Ours)†	2.81	84	75	64	3.13	81	73	62

Figure from “BEVBert: Multimodal Map Pre-training for Language-guided Navigation” (2023)

As a reference, the navigation Success Rate (SR) for a human navigator is 86%.

The VLN problem involves many different areas of AI



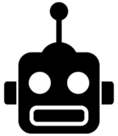
Calculating Cross-attention

Learning to Navigate Unseen Environments: Back Translation with Environmental Dropout (Tan et al. 2019)

History aware multimodal transformer for vision-and-language navigation (Chen et al. 2021)

Think Global, Act Local: Dual-Scale Graph Transformer for Vision-and-Language Navigation (Chen et al. 2022)

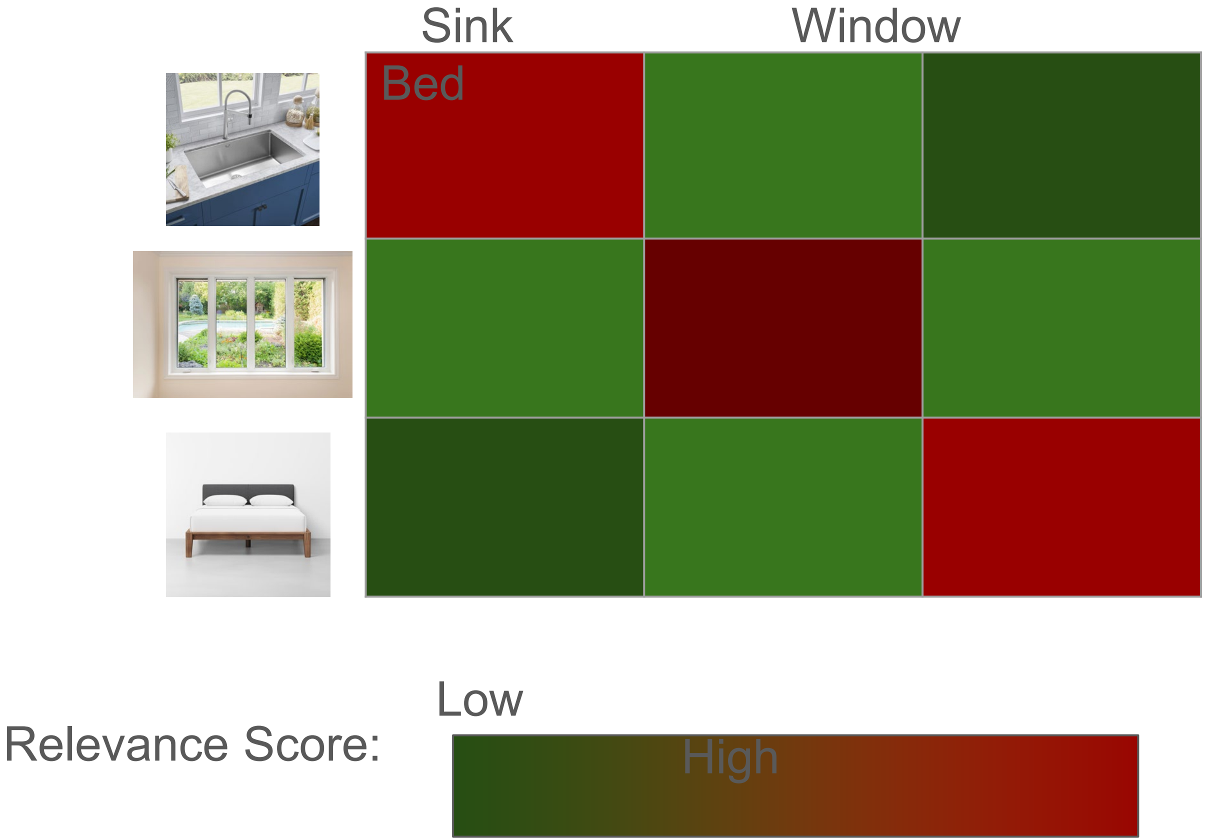
One of the most important task for a VLN system to learn is how to relate images to text fragments in the instruction:



"Turn left from the **window**, walk away from the **sink**, go into the bedroom. Go to the **bed**."

This is important in VLN, because these images are the views that represent the available direction to go in. So relating view & instruction ≈ relating action & instruction.

Attention mechanism: to enable the VLN system to **focus** on a certain image based on its **relevance** to a certain text fragment (instruction).



This is also helpful between text or images themselves

Soft-Dot attention: “Env-Drop”, Tan et al., 2019

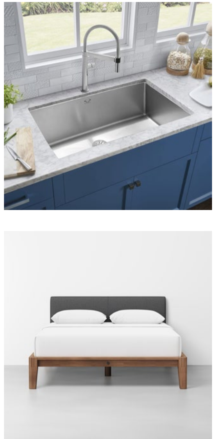
How to focus: when making decision, give relevant subject(s) **softmax’ed** higher weights compared to other ones.

How to calculate relevance: similarity, e.g. **dot-product**.

Say, given the instruction “go near the bed”, what we want is to let the agent head to the direction represented by the image of a bed instead of a sink.

“bed” 1 x 512

512 x 2



Dot-product (matrix mul)

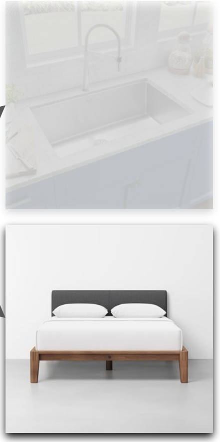
Softmax

$$s(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$

Attention score

1 x 2

[0.1,0.9]



0.1

0.9

(I should go this way)

Attention by Transformer (in HAMT, Chen et al. 2021)

Query vector: 1 x 512

Key vector 1 x 512

"bed" → $Q_{\text{``bed''}}$, $K_{\text{``bed''}}$

→ Q_{bed}, K_{bed}

→ Q_{sink}, K_{sink}

Step 1:
Get attention

$$Q_{\text{``bed''}} \times K_{\text{bed}}^T$$
$$Q_{\text{``bed''}} \times K_{\text{sink}}^T$$

attn_{bed}
$\text{attn}_{\text{sink}}$

Step 2:
Get normalized attention score (in a form of "probability")

$$\text{softmax}\left(\frac{\text{attn}_{\text{bed}}}{\sqrt{d}}, \frac{\text{attn}_{\text{sink}}}{\sqrt{d}}\right)$$

prob_{bed}
$\text{prob}_{\text{sink}}$

In theory, these attention scores can be used to decide which way to go.
Nonetheless, practically the navigation action prediction is based on the "contextualized" features, instead of just the normalized attention scores.
Remember that the action is taken based on the images that represent the direction to go, so the attention scores we are more curious about are 1 image v.s. K words instead 1 word v.s. K images

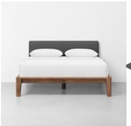
Attention by Transformer (in HAMT, Chen et al. 2021)

E.g., we want to predict how likely the agent should go to the direction with a bed object in its represented view, given the instruction (context) “go bed”.

Query vector: 1 x 512

Key vector 1 x 512

Value vector 1 x 512



Q_{bed}, K_{bed}, V_{bed}

“go” $Q_{\text{“go”}}$, $K_{\text{“go”}}$, $V_{\text{“go”}}$

“bed” $Q_{\text{“bed”}}$, $K_{\text{“bed”}}$, $V_{\text{“bed”}}$

Step 1:
Get attention

$$Q_{\text{bed}} \times K^{T}_{\text{“go”}}$$
$$Q_{\text{bed}} \times K^{T}_{\text{“bed”}}$$

$\text{attn}_{\text{“go”}}$
$\text{attn}_{\text{“bed”}}$

Step 2:
Get normalized attention score (in a form of “probability”)

$$\text{softmax}\left(\frac{\text{attn}_{\text{“go”}}}{\sqrt{d}}, \frac{\text{attn}_{\text{“bed”}}}{\sqrt{d}}\right)$$

$\text{prob}_{\text{“go”}}$
$\text{prob}_{\text{“bed”}}$

Step 3:
Multiply normalized attention scores with the value vector

$$\text{prob}_{\text{“go”}} \cdot V_{\text{“go”}}$$
$$\text{prob}_{\text{“bed”}} \cdot V_{\text{“bed”}}$$

Step 4:
Sum them all up to be the **new feature** for the view **bed**” (but now it is “**contextualized**” by the instruction “go bed”.

$$\text{prob}_{\text{“go”}} \cdot V_{\text{“go”}} + \text{prob}_{\text{“bed”}} \cdot V_{\text{“bed”}}$$

1 x 512

FC

Contextualized Bed Features

Full Connected layer

And you can repeat this process to calculate the action score for other available actions with different view images.

softmax

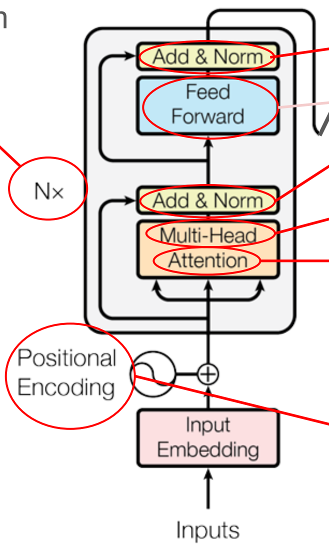
prob_{bed}

Attention by transformer (in HAMT, Chen et al. 2021)

With the core attention mechanic explained, we can now build a “full” transformer encoder with some extra techniques.

Encoder stacking: we can stack them up multiple times one after another.

Contextualized features



Layer normalization & residual Layer: provide stability in updated features, alleviate vanishing gradients, etc.

FeedForward: just linear activation layer, nothing special

Multiple head: allowing multiple set of relationship between the same image-text pairs.

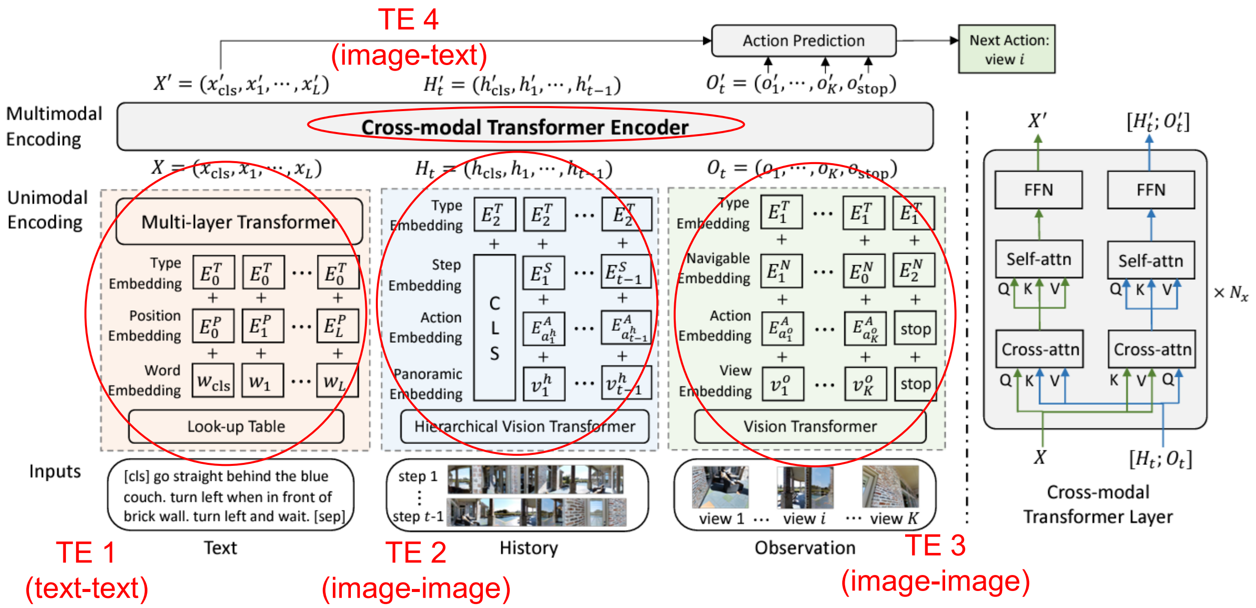
The attention calculation

Positional encoding: for text, this additional embedding (encoding) indicates the position of the word in the instruction. Similarly, positional encoding can be applied to images regarding direction, angle… as well.

Attention by transformer (in HAMT, Chen et al. 2021)

Remember that Transformer Encoder (TE) works on tokens/embeddings regardless of whether they are images or texts. That means the transformer encoder can calculates contextualized texts from texts (Machine Translation), images from images (ViT for classification) as well.

The HAMT model



Attention by transformer (in RecBERT, Hong et al. 2021)

(Each word including [CLS] is a token, represented by a feature vector, just like an image)

"**[CLS (step 1)]** Turn left from the **window**, walk out of the **sink**, go into the bedroom. Go to the **bed**."

(Done)

"**[CLS (step 2)]** Turn left from the **window**, walk out of the **sink**, go into the bedroom. Go to the **bed**."

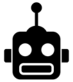
(Done)

"**[CLS (step 3)]** Turn left from the **window**, walk out of the **sink**, go into the bedroom. Go to the **bed**."

[CLS (step 3)] →

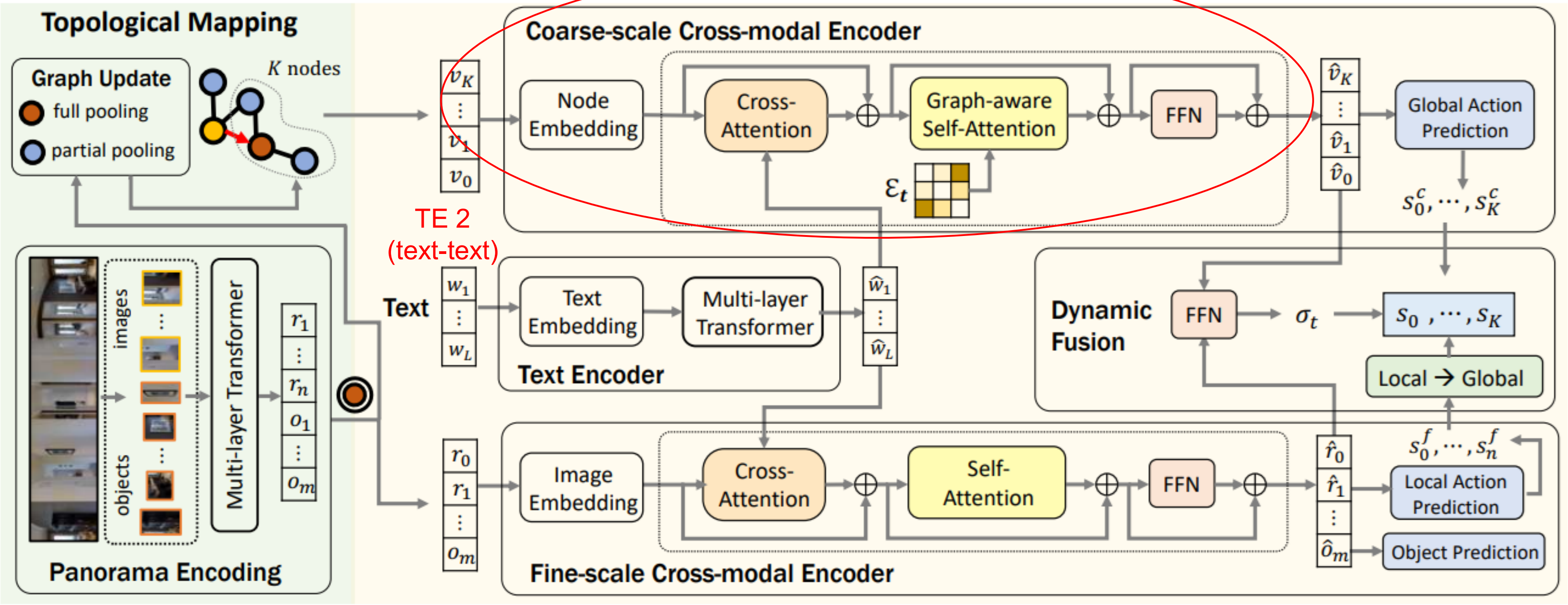
$\mathrm{prob}_{\mathrm{bed}} \uparrow$

(I should go this way)



Attention by transformer (in DUET, Chen et al. 2022)

TE 1
(viewpoint-viewpoint)



TE 3
(image+object)-
(image+object)

Data Augmentation w/ multimodal learning

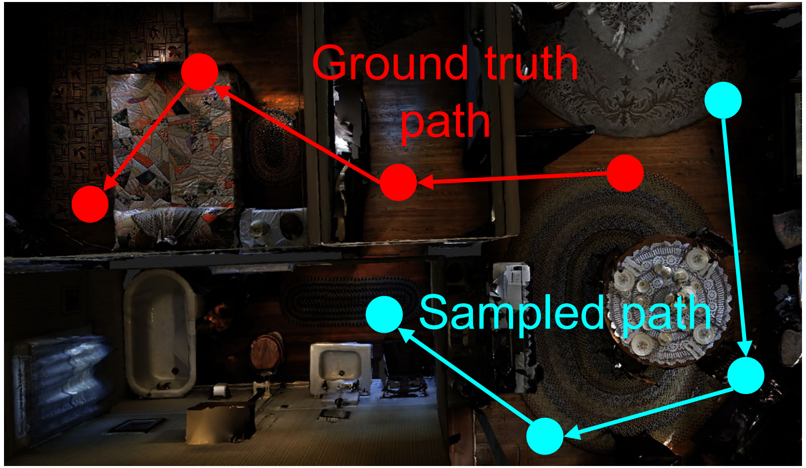
Speaker-Follower Models for Vision-and-Language Navigation (Fried et al. 2018)

Less is More: Generating Grounded Navigation Instructions from Landmarks (Wang et al. 2022)

PanoGen: Text-Conditioned Panoramic Environment Generation for Vision-and-Language Navigation (Li et al. 2023)

The navigation data is expensive to collect, especially those from the real life. Data augmentation is a way to train a model to generalize better without the need of additional labor to collect data.

Data Augmentation by Generating More Instruction for sampled Path (in Speaker & Follower from Fried et al. 2018)

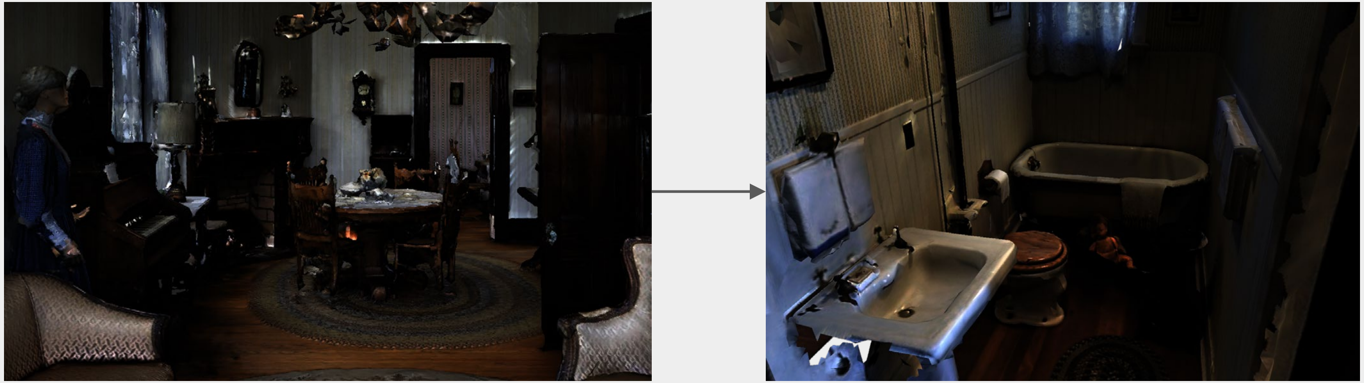


Human annotator

"Walk into the room, go inside another room again, stop on the other side of the bed."

7,189 paths
21.5k instructions
Roughly the same amount of data as the original dataset.

Image-to-text sequential model (Seq2Seq)



"Walk across the table and get into the bathroom, stop."

Data augmentation by Landmark Alignment (in Marky, Wang et al. 2022)

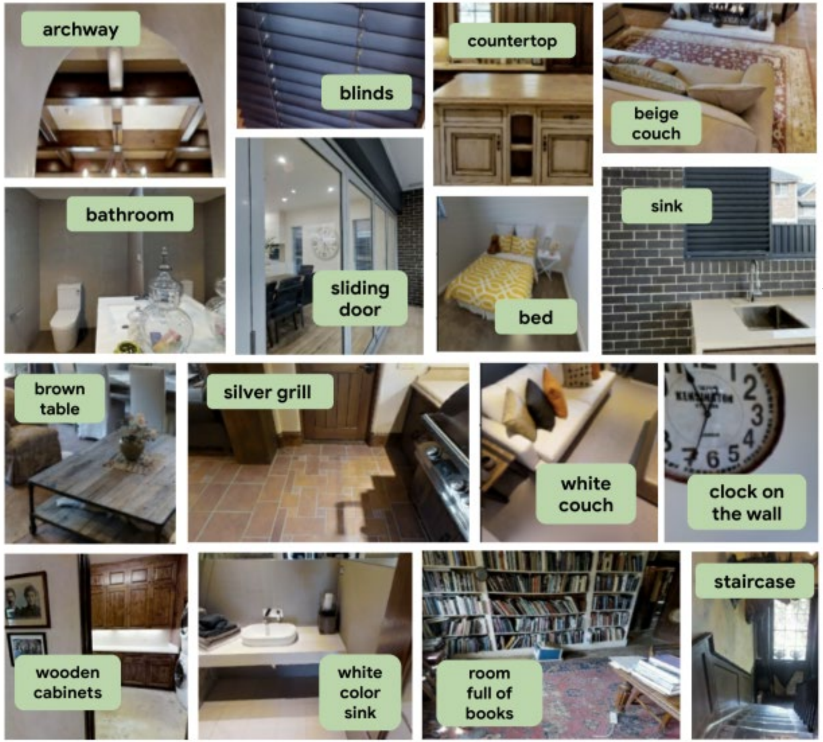
If we think about what an instruction is made of given any navigation path, mainly there are two types of information decide how we describe it:

1. Camera view and its change over steps
2. Landmark of the view

E.g., You're starting in a laundry room, facing the railing. Walk out of this laundry room onto the wooden flooring. Turn right and go down the hallway toward the end of this hallway.

To generate any instruction for a path, we need to fill-in the two types of information above. We can describe the change of the camera view by calculate the angle between “inbound” and “outbound” given the trajectory of the path (x-y-z axis). So what’s “tricky” is how do we locate the seen “landmarks” within the view images.

Data augmentation by Landmark Alignment (in Marky, Wang et al. 2022)



The idea is to

1. **build an landmark alignment model that can detect/recognize visual landmarks along the navigation path.**
2. Given a scene(navigation environment), detect landmarks in it.

Data augmentation by Landmark Alignment (in Marky, Wang et al. 2022)

You are standing in front of a ***brown chair****. Take a left to enter the* ***bathroom****, you will see a* ***sink*** *in front of you. Now take a step to the right at stop at the* ***foot mat****, you will have reached your destination.*
→ [***brown chair****,* ***bathroom****,* ***sink****,* ***foot mat***].

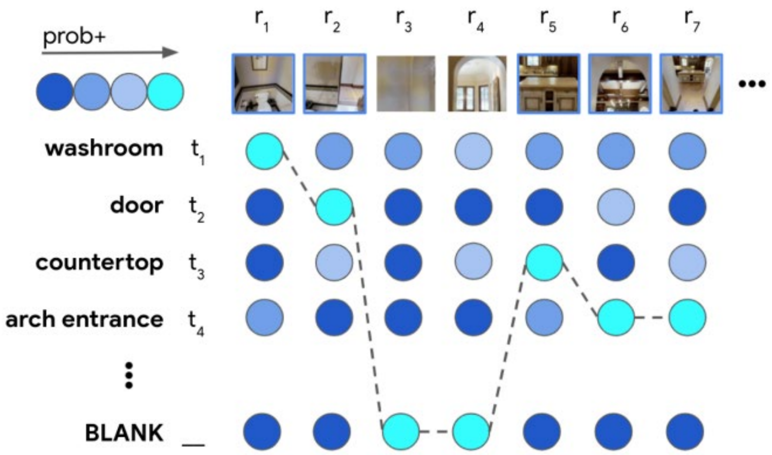
Visual Candidates
(Potentially Landmarks)

Textual
Landmarks

Align them

In this case, we obtain the real landmarks from all images we perceive during the path.

Data augmentation by Landmark Alignment (in Marky, Wang et al. 2022)



Relevance between any image-text pair:

$$A_{i,j} = X(t_i) \cdot Y(r_j) - \lambda(T(t_i) - T(r_j))^2$$

Text-Image feature similarity

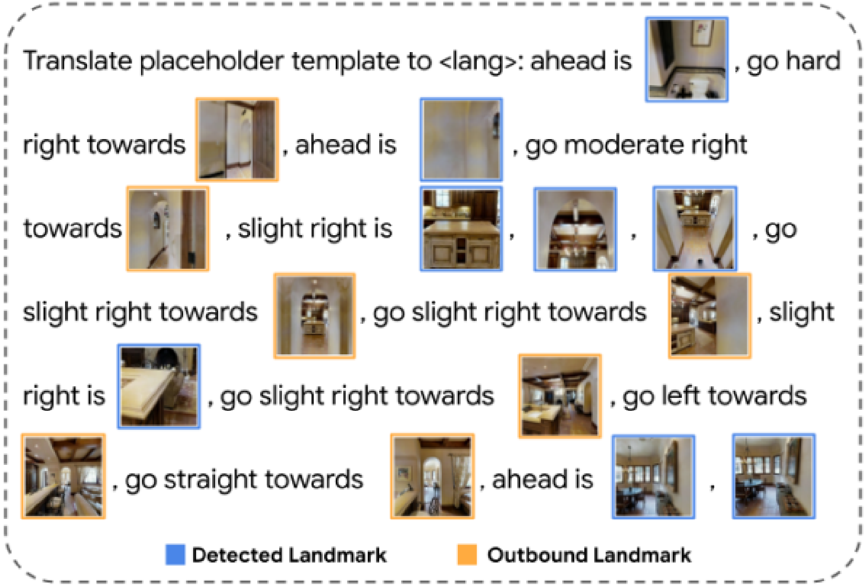
Temporal difference

Connectionist Temporal Classification (CTC) loss:

$$p(\boldsymbol{t} \mid \boldsymbol{r}) = \sum_{A^{(\boldsymbol{r},\boldsymbol{t})}} \prod_{j=1}^{n} p(A_{i,j} \mid r_j)$$

Find out the most likely alignment between the set of images and the set of landmark texts among all the possible alignments.

Data augmentation by Landmark Alignment (in Marky, Wang et al. 2022)



You are facing towards the commode. Turn right and exit the washroom. Turn right and walk straight till you reach the white cabinet in the front. There is an arch in the front. Enter inside the arch. Turn right and walk towards the sofa. Turn left and walk straight till you reach the arch in the front. There is a round table with four chairs towards your left side. You have reached your point.

The idea is to (continued):
3. Generate a path connecting the sampled landmarks.
4. Generate the instruction by the landmarks and description of the camera view change.

Result: 1 million+ navigation instructions as “silver data”.
(The term “silver data” refers to high-quality annotations–not created by people–that are derived by combining models and constraints)

Data augmentation by generating more scenes in various style (in PanoGen, Li et al. 2023)

The biggest challenge in VLN is to understand scenes that have never been seen during training.
E.g., even for the same instruction “go to the bedroom”, views that the agent sees can be vastly different between training and testing (especially in terms of style).

“bedroom”

Data augmentation by generating more scenes in various style (in PanoGen, Li et al. 2023)

The way to augment the existing data for PanoGen is based on the opposite way to the works previously discussed, i.e., to generate images given specific texts.

The specific texts are the description of the scene (caption), and PanoGen generates images based on the same description but in a different “style”.

Captioning model (BLIP-2)

“a bedroom with a bed”

Generative model (Stable Diffusion)

But there’s an issue that, VLN models take a set of discretized images from a panoramic view as visual input. So the captioning model generates description based on these discretized images instead of the panoramic view.

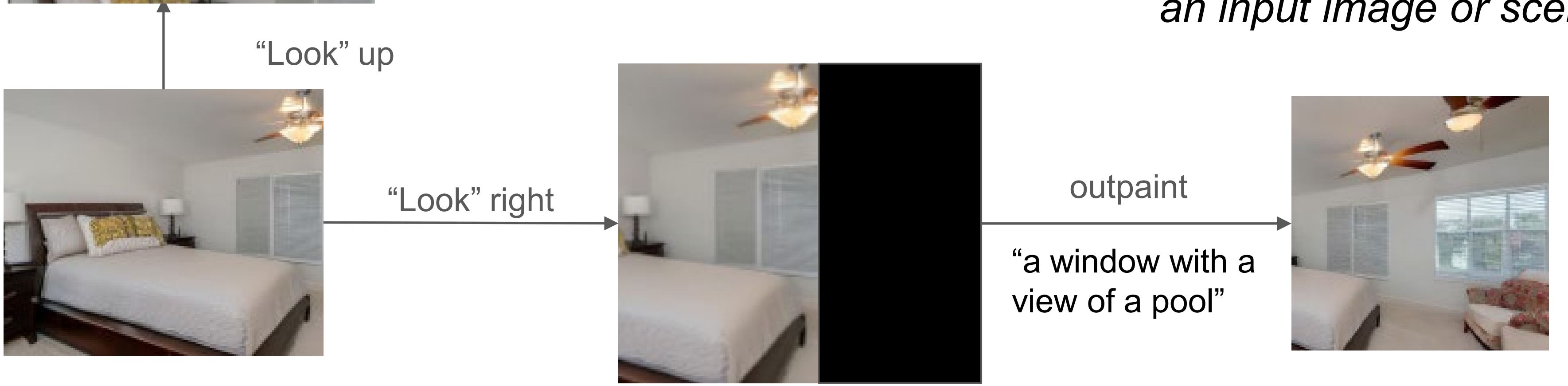
Data augmentation by generating more scenes in various style (in PanoGen, Li et al. 2023)

The generated images look okay individually. But they do not make up a “continuous” panoramic view (incoherent panorama).

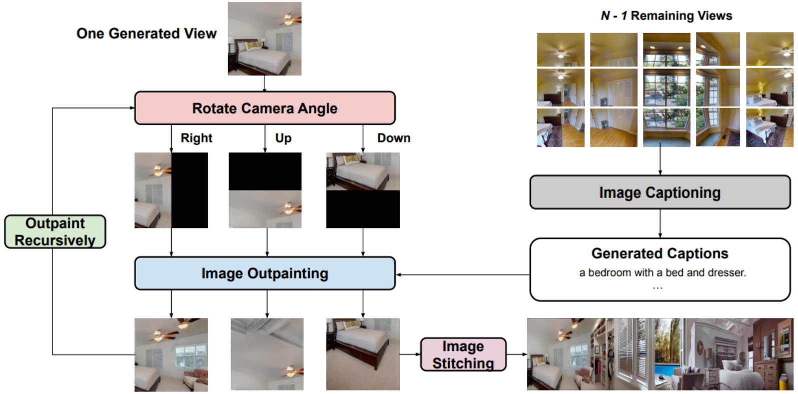
Data augmentation by generating more scenes in various style (in PanoGen, Li et al. 2023)

Instead, PanoGen asked the Stable Diffusion model to generate the panoramic view recursively

("Outpainting" in the context of generative models refers to the task of generating content beyond the boundaries of an input image or scene.)

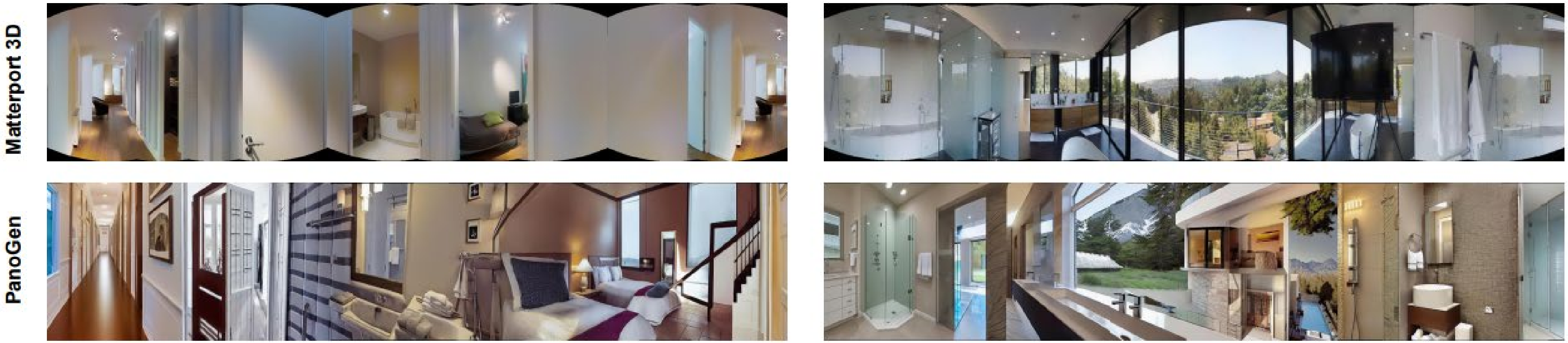


Data augmentation by generating more scene views in various style (in PanoGen, Li et al. 2023)



By recursive outpainting, the panoramic view from stitched the generated images has better coherence than generating them separately.

7644 panoramas
Replacing 30% of the panoramas during VLN model fine-tuning.



Summary

1. Vision-and-Language Navigation
   a. Definition
   b. Related Research Areas
2. Multimodal Attention:
   a. Soft-dot attention (Env-Drop)
   b. Transformer encoder attention (HAMT & DUET)
3. Data Augmentation
   a. Instruction Augmentation: Speaker (& Follower) model
   b. Instruction Augmentation: Marky
   c. View Augmentation: PanoGen