# Multimodal Text Style Transfer for Outdoor Vision-and-Language Navigation

Wanrong Zhu[1], Xin Wang[2], Tsu-Jui Fu[1], An Yan[3], Pradyumna Narayana[4], Kazoo Sone[4], Sugato Basu[4], and William Yang Wang[1]

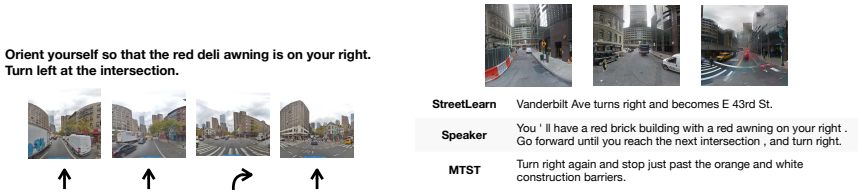
[1] University of California, Santa Barbara
{wanrongzhu, tsu-juifu}@ucsb.edu, william@cs.ucsb.edu
[2] University of California, Santa Cruz xwang366@ucsc.edu
[3] University of California, San Diego ayan@eng.ucsd.edu
[4] Google {pradyn, sone, sugato}@google.com

**Abstract.** In the vision-and-language navigation (VLN) task, an agent follows natural language instructions and navigate in visual environments. Compared to the indoor navigation task [2] that has been broadly studied, navigation in real-life outdoor environments such as Touchdown [5] remains a significant challenge with its complicated visual inputs and an insufficient amount of instructions that illustrate the intricate urban scenes. In this paper, we introduce a Multimodal Text Style Transfer (MTST) framework to mitigate the data scarcity problem in outdoor navigation tasks by effectively leveraging external multimodal resources. We first enrich the navigation data by transferring the style of the instructions generated by Google Map API, then pre-train the navigator with the augmented external outdoor navigation dataset. Experimental results show that our MTST approach is model-agnostic, and our MTST framework significantly outperforms the baseline models on the outdoor VLN task, improving task completion rate by 22% relatively and achieving new state-of-the-art performance.

**Keywords:** Vision-and-language navigation, multimodal style transfer.

## 1 Introduction

Vision-and-language navigation (VLN) is a task where an agent navigates in a real environment by following natural language instructions, as illustrated in Fig. 1a. Different from indoor navigation [2, 32, 9, 33, 22, 29, 23, 18], the outdoor navigation task [5, 25, 24] takes place in urban environments that contain diverse street views. The vast urban area leads to a much larger space for an agent to explore, which provides a wide variety of objects for visual grounding and requires more complicated instructions to address the sophisticated navigation environment. It is also more difficult to recover from a mistaken action when routing in a real-life urban environment. These problems made outdoor navigation a much more challenging task than indoor navigation, which has been broadly studied. Although tools such as Google Map API enable researchers




(a) An example of navigation in outdoor environment.

(b) The comparison between instructions from the speaker model and our MTST.

Fig. 1: The example of outdoor vision-and-language navigation and multimodal text style transfer.

to gather large-scale street scenes for visual perception, it is expensive to collect human-annotated instructions and generate adequate trajectory-instruction pairs to train an agent. The issue of data scarcity limits navigation performance under sophisticated urban environments.

To deal with the data scarcity issue, Fried et al. [9] proposes a Speaker model to generate additional training pairs. By sampling plenty of trajectories in the environment and adopting the Speaker model to back-translate their instructions, one can obtain a broad set of augmented training data.

Although pre-training with augmented data produced by the Speaker model improves the agent's performance to some extent [9], there are a few inherent drawbacks of the method that limits the benefits of introducing augmented trajectories. First, the trained Speaker model hardly back-translates specific objects in the trajectory — it can only provide general guidance regarding directions (Fig. 1b). As a result, it is challenging to learn the alignment between language and object groundings with the augmented data generated by the Speaker model. Moreover, since the back-translated instructions are reconstructed by a Speaker model, we can not assure its correctness — any error within the augmented instructions will propagate into the navigation agent during the pre-training process and hinder the final navigation performance. Thus, Speaker-based data augmentation mainly relies on the Speaker model's reconstruction, but the unstable quality issue massively decreases its effectiveness.

To overcome data scarcity but avoid the error propagation issue, we leverage external resources to help outdoor navigation. Google Street View[5] has world-wide scale coverage of street scenes, which also supplies the navigation service between two locations. With it, we can collect various additional navigation trajectories in the urban environment. One of the challenges of utilizing external resources is the style distinction between instructions. For example, human-annotated instructions in the original outdoor navigation task emphasize attributes of the visual environment as navigation targets. As for the external trajectory provided by Google Street View, the instructions are generated by Google Maps API[1], with the street- and direction-guidance only. Though provid-

[5] https://developers.google.com/maps/documentation/streetview/intro

ing trajectory-instruction pairs, the instruction's style distinction will decrease the power of data augmentation.

In this paper, we present a novel multimodal text style transfer (MTST) framework that can leverage external resources to overcome the data scarcity issue in the outdoor VLN task while addressing the style distinction between instructions, as shown in Fig. 1b. To maximize the effectiveness of introducing external augmented data, we apply the trained multimodal style-transfer model to transfer the instructions from the external resources into our original style. Then, the style-transferred instructions will serve as augmented data to pre-train the navigation agent. Under this approach, we can utilize the exact instructions from the external resources, avoiding the error propagation issue from the Speaker model. Moreover, in addition to the direction-guided information, the mentioned objects which are recovered by our multimodal style-transfer model can help learn the grounding between navigation targets and visual environment. The additional style-transferred instructions, which are style-corresponding to the original task, can increase the robustness of the pre-training process and solve the data scarcity issue in the outdoor navigation. Meanwhile, the injection of the new environment from external resources can improve the navigation agent's generalizability.

Experimental results show that leveraging external resources during the pre-training process improves the navigation agent's performance. In addition, the more style-corresponding instructions that are transferred by our multimodal style-transfer model can bring out more improvement and make the pre-training process more robust. In summary, the contribution of our work is three-fold:

- We leverage external multimodal resources into the outdoor VLN task to overcome the data scarcity issue, avoiding the error propagation problem from the Speaker model.
- We introduce a novel multimodal text style transfer framework to transfer the instructions from the external resources into our original style and make more robust augmented data, which leads to better navigation performance.
- Apart from the empirical results, we also demonstrate that our multimodal style-transfer model indeed produces style-corresponding instructions for the external resources — hence the navigator can benefit during the pre-training process.

## 2 Related Work

**Vision-and-Language Navigation** Vision-and-language navigation (VLN) [5, 24, 20, 2] is a task that requires an agent to achieve the final goal based on the given instructions in a 3D environment. Besides the generalizability problem studied by many previous works on the VLN task [32, 9, 33, 22, 29, 23, 18], the data scarcity problem is still a critical issue for the VLN task. For the outdoor navigation task [5, 24], since the instructions are annotated by humans, it is difficult to collect large-scale human-written data for training agents under

vast urban environments. This kind of data scarcity makes learning the optimal match between vision-and-language challenging. In this paper, we introduce the multimodal text-style transfer model that can utilize additional street view scenes and leverage different domains' instructions to further improve the outdoor navigation agent.

**Pre-training for Vision-and-Language Navigation** To deal with the data scarcity problem, Fried et al. [9] proposes the Speaker model. The Speaker model is trained from the original trajectory-instruction pairs and back-translates the instruction of a navigation trajectory. By sampling numerous trajectories with the reconstructed instructions in the environment, the navigation agent can pre-train for the augmented data [10, 12]. Though these pre-training methods show some improvement, they mainly rely on the quality of the instructions reconstructed from the Speaker model. However, the Speaker model can only produce instructions with guidance on direction and hardly back-translate specific target objects. This restriction heavily limits the benefit of introducing augmented trajectories by the Speaker model. For our proposed multimodal text style transfer model, instead of being wholly based on the Speaker model, we take advantage of numerous yet high-quality instructions from a different domain and apply style-transfer for a better pre-training process.

**Leveraging External Resources** With the numerous but unlabeled data from external resources, effectively utilizing them can improve performance and generalizability. Chen et al. [4] searches related articles on Wikipedia to answer open-domain questions. Wang et al. [34] leverages the external corpus on WikiHow to realize zero-shot video captioning. Zheng et al. [37] helps neural machine translation by incorporating the information from human interactive suggestions. In this paper, we utilize the urban environment resources from Google Street View to enhance the outdoor navigation task. Instead of leveraging them directly, we propose a multimodal style-transfer model to make the external instructions more robust to our primary task.

**Style Transfer for Data Augmentation** Data augmentation is a training technique that avoids the trained model from overfitting and enhances its generalizability by generating more diverse data as training input [6]. Unlike traditional data augmentation methods like rotating, flipping, and cropping the image, style transfer [11] can simultaneously maintain the original semantic and supply more distinct image features for data augmentation. Jackson et al. [16] adopts style transfer between two domains to increase the model's robustness for cross-domain image classification. Xu and Goel [36] also proposes domain adaptive text style transfer to leverage massively available text data from other domains. Inspired by the above data augmentation methods via style transfer, our multimodal style transfer utilizes cross-domain adaption for augmented data and realizes instruction recovery with the reference trajectory to enhance the navigation agent.

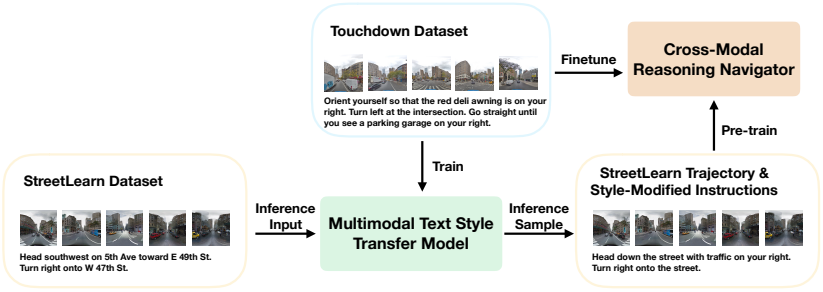



Fig. 2: An overview of the multimodal text style transfer (MTST) framework for vision-and-language navigation in real-life urban environments.

## 3 Methodology

### 3.1 Background

**Vision-and-Language Navigation (VLN)** In the vision-and-language navigation task, the reasoning navigator is asked to find the correct path to reach the target location following the guidance of the instructions (a set of sentences) $\mathcal{X} = \{s_1, s_2, \ldots, s_m\}$. The navigation procedure can be viewed as a series of decision making processes. At each time step $t$, the navigation environment presents an image view $\boldsymbol{v}_t$. With reference to the instruction $\mathcal{X}$ and the visual view $\boldsymbol{v}_t$, the navigator is expected to choose an action $a_t \in \mathcal{A}$. The action set $\mathcal{A}$ for urban environment navigation usually contains four actions, namely *turn left*, *turn right*, *go forward*, and *stop*.

**Instruction Style** The navigation instructions vary across different vision-and-language navigation datasets in real-world urban environments. The instructions in the StreetLearn dataset [25], a large-scale interactive navigation environment built upon Google Street View[6], is generated by Google Maps API. Street names constantly appear in the templated-based instructions in StreetLearn. In contrast, the human-written instructions in the Touchdown dataset [5, 24], another urban navigation environment with Street View panoramas, frequently refer to objects in the panorama, such as traffic lights, cars, awnings, etc. Our multimodal style-transfer framework can effectively utilize the external data from the StreetLearn dataset, and leverage the instructions with different styles.

### 3.2 Overview

Following natural language instructions and navigating through a busy urban environment, as proposed in the Touchdown task [5], remains a great challenge for navigation agents. In this paper, we propose the Multimodal Text Style Transfer (MTST) framework for the vision-and-language navigation in real-life urban environments. The MTST framework mainly consists of two modules,

[6] https://developers.google.com/maps/documentation/streetview/intro

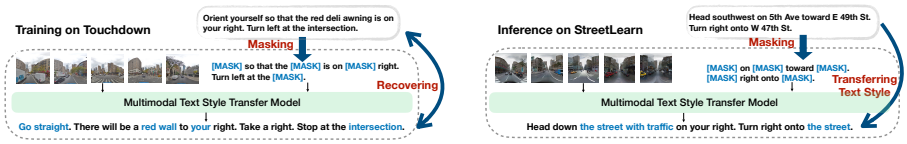



Fig. 3: An overview of the training and inference process of the multimodal text style transfer model. In the training process, with object-related tokens in Touchdown instructions being masked out, the model is encouraged to recover the instruction with the help of the incomplete instruction template and the visual features in the trajectory. When inferencing new instructions for StreetLearn trajectories, we mask the street names in the original instructions and prompt the model to generate new instructions in Touchdown style.

namely the **multimodal text style transfer model** and the **cross-modal reasoning navigator**. Fig. 2 provides an overview of our MTST framework.

To mitigate the data scarcity problem in the Touchdown dataset, we leverage the StreetLearn dataset in our training process. We use the multimodal text style transfer model to narrow the gap between the two datasets' different domains, especially the significant difference between the navigation instructions. Trained on the Touchdown dataset, the MTST model learns to infer style-modified instructions for StreetLearn trajectories.

Furthermore, we apply the two-stage training pipeline to train the cross-modal reasoning navigator. We first pre-train our navigator on the StreetLearn trajectories paired with style-modified instructions, then fine-tune the navigator on the Touchdown dataset.

### 3.3 Multimodal Text Style Transfer Model

In this section, we introduce the detailed implementation of the multimodal text style transfer model.

As can be seen in Fig. 2, the main difference between Touchdown instructions and StreetLearn instructions is that the human-written instructions in Touchdown focus on objects in the surrounding environment, while the machine-generated instructions in StreetLearn emphasize on street names nearby. The goal of conducting multimodal text style transfer is to inject more object-related information in the surrounding navigation environment to the StreetLearn instruction while keeping the correct guiding signals.

**Mask-and-Recover Scheme** To inject objects that appeared in the panorama into the instructions, the multimodal text style transfer model is trained with a "masking-and-recovering" scheme. We train the model on the Touchdown dataset, then infer instructions for trajectories in the StreetLearn dataset. We mask out certain portions in the instructions and try to recover the missing portions with the help of the remaining instruction skeleton and the paired trajectory. To be specific, we use NLTK [3] to mask out the object-related tokens in the Touchdown instructions, and the street names in the StreetLearn instruc-

tions[7]. Multiple tokens that are masked out in a row will be replaced by a single [MASK] token. We aim to maintain the correct guiding signals for navigation after the style transfer process. Tokens that provide guiding signals, such as "turn left" or "take a right", will not be masked out. Instead, they will be part of the remaining instruction skeleton that the style transfer decoder will attend to when generating instructions with the new style.

Fig. 3 provides an example of the "mask-and-recover" process during training and inferencing. The MTST model is trained on the Touchdown dataset. We mask out the objects in the human-annotated instructions to get the instruction template, and the training objective is to generate an instruction that recovers the missing objects given the visual trajectory and the masked instruction template. With the "mask-and-recover" training scheme, the MTST model learns to generate object-grounded instructions. When inferring instructions for the StreetLearn dataset, the MTST model also takes the visual trajectory and the masked instruction template as input. However, the masked instruction template is generated from the instructions provided in Google Map API, and it is transferred to the human-annotated and object-grounded style by our MTST model. Those generated instructions will be used to pretrain the navigator.

**Model Structure** We build our multimodal text style transfer model based on the Speaker model, as proposed by Fried et al. [9]. On top of the visual-attention-based LSTM [14] structure in the Speaker model, we inject the textual attention of the masked instruction skeleton $\mathcal{X}'$ to the encoder, which allows the model to attend to original guiding signals.

The encoder takes both the visual and textual inputs, which encodes the trajectory and the masked instruction skeletons, respectively. To be specific, each visual view in the trajectory is represented as a feature vector $\boldsymbol{v}' = [\boldsymbol{v}'_v; \boldsymbol{v}'_\alpha]$, which is the concatenation of the visual encoding $\boldsymbol{v}'_v \in \mathbb{R}^{512}$ and the orientation encoding $\boldsymbol{v}'_\alpha \in \mathbb{R}^{64}$. The visual encoding $\boldsymbol{v}'_v$ is the output of the last but one layer of the RESNET18 [13] of the current view, while the orientation encoding $\boldsymbol{v}'_\alpha$ encodes current heading $\alpha$ by repeating vector $[sin\alpha, cos\alpha]$ for 32 times. As described in section 3.4, the feature matrix of a panorama is the concatenation of eight projected visual views.

In the multimodal style transfer encoder, we use a soft-attention module [30] to calculate the grounded visual feature $\hat{\boldsymbol{v}}_t$ for current view at step $t$:

$$attn_{v_{t,i}} = softmax((\boldsymbol{W}_v \boldsymbol{h}_{t-1})^T \boldsymbol{v}'_i),\ \hat{\boldsymbol{v}}_t = \sum_{i=1}^{8} = attn_{v_{t,i}} \boldsymbol{v}'_i \tag{1}$$

where $\boldsymbol{h}_{t-1}$ is the hidden context of previous step, $\boldsymbol{W}_v$ refers to the learnable parameters, and $attn_{v_{t,i}}$ is the attention weight over the $i_{th}$ slice of view $\boldsymbol{v}'_i$ in current panorama.

The textual encoding $\boldsymbol{s}'$ for each masked sentence in the instruction skeleton is the average of the word embeddings. We also use a soft-attention modules to

[7] We masked out the tokens with the following part-of-speech tags: [JJ, JJR, JJS, NN, NNS, NNP, NNPS, PDT, POS, RB, RBR, RBS, PRP$, PRP, MD, CD]

calculate the grounded textual feature $\hat{\boldsymbol{s}}_t$ at current step $t$:

$$attn_{s_{t,j}} = softmax((\boldsymbol{W}_s \boldsymbol{h}_{t-1})^T \boldsymbol{s}'_j),\ \hat{\boldsymbol{s}}_t = \sum_{j=1}^{M} attn_{s_{t,j}} \boldsymbol{s}'_j \tag{2}$$

where $\boldsymbol{W}_s$ refers to the learnable parameters, $attn_{s_{t,j}}$ is the attention weight over the $j_{th}$ masked sentence encoding $\boldsymbol{s}'_j$ at step $t$, and $M$ denotes the maximum sentence number in the masked instruction skeleton $\mathcal{X}'$.

Based on the grounded visual feature $\hat{\boldsymbol{v}}_t$, the grounded textual feature $\hat{\boldsymbol{s}}_t$ and the visual view feature $\boldsymbol{v}'_t$ at current timestamp $t$, the hidden context can be given as:

$$\boldsymbol{h}_t = LSTM([\hat{\boldsymbol{v}}_t; \hat{\boldsymbol{s}}_t; \boldsymbol{v}'_t]) \tag{3}$$

**Training Objectives** We train the multimodal text style transfer model in the teacher-forcing manner [35]. The decoder generates tokens auto-regressively, conditioning on the masked instruction template $\mathcal{X}'$, and the trajectory.

The training objective is to minimize the following cross-entropy loss:

$$\mathcal{L}(x_1, x_2, \ldots, x_n | \mathcal{X}', \boldsymbol{v}'_1, \ldots, \boldsymbol{v}'_N) = -\log \prod_{j=1}^{n} P(x_j | x_1, ..., x_{j-1}, \mathcal{X}', \boldsymbol{v}'_1, \ldots, \boldsymbol{v}'_N) \tag{4}$$

where $n$ is the total token number in original instruction $\mathcal{X}$, and $N$ denotes the maximum view number in the trajectory.

### 3.4 Cross-Modal Reasoning Navigator

In this section, we introduce the implementation details of the Cross-Modal Reasoning Navigator(CMRN). As illustrated in Fig. 4, our reasoning navigator is composed of an instruction encoder, a trajectory encoder, a cross-modal Transformer that fuses the modality of the instruction encodings and trajectory encodings, and an action predictor.

**Instruction Encoder** We use the instruction encoder to generate embeddings for each sentence $s_i$ in the instruction $\mathcal{X}$.

The instruction encoder is a pre-trained uncased BERT-base model[7]. For the $i_{th}$ sentence $s_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,l_i}\}$ that contains $l_i$ tokens, its sentence embedding $\boldsymbol{h}_i^s$ is calculated as:

$$\boldsymbol{w}_{i,j} = \mathcal{BERT}(x_{i,j}) \in \mathbb{R}^{768},\ \boldsymbol{h}_i^s = \mathcal{FC}(\frac{\sum_{j=1}^{l_i} \boldsymbol{w}_{i,j}}{l_i}) \in \mathbb{R}^{256} \tag{5}$$

where $\boldsymbol{w}_{i,j}$ is the word embedding for $x_{i,j}$ generated by BERT, and $\mathcal{FC}$ is a fully-connected layer.

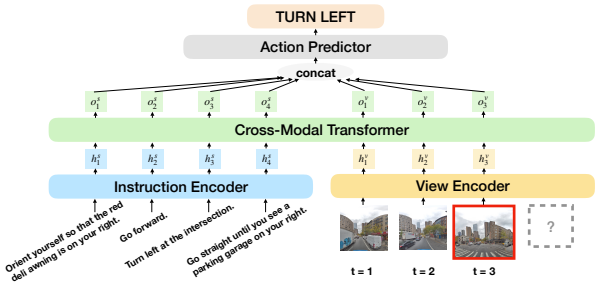



Fig. 4: The overall structure of our cross-modal reasoning navigator. In this example, the navigator predicts to take a left turn for the visual scene at $t = 3$.

**View Encoder** We use the view encoder to retrieve embeddings for the visual views at each time step.

Following Chen et al. [5], we embed each panorama $\mathbf{I}_t$ by slicing it into eight images and projecting each image from an equirectangular projection to a perspective projection. Each of the projected image of size $800 \times 460$ will be passed through the RESNET18 [13] pre-trained on ImageNet [28]. We use the output of size $128 \times 100 \times 58$ from the fourth to last layer before classification as the feature for each slice. The feature map for each panorama is the concatenation of the eight image slices, which is a single tensor of size $128 \times 100 \times 464$.

We center the feature map according to the agent's heading $\alpha_t$ at time stamp $t$. We crop a $128 \times 100 \times 100$ sized feature map from the center and calculate the mean value along the channel dimension. The resulting $100 \times 100$ features is regard as the current panorama feature $\hat{\mathbf{I}}_t$ for each state. Following Mirowski et al. [25], we then apply a three-layer convolutional neural network on $\hat{\mathbf{I}}_t$ to extract the view features $\boldsymbol{h}_t^v \in \mathbb{R}^{256}$ at time stamp $t$.

**Cross-Modal Transformer** In order to navigate through complicated real-world environments, the agent needs to grasp a proper understanding of the natural language instructions and the visual views jointly to choose proper actions for each state. Since the instructions and the trajectory lies in different modalities and are encoded separately, we introduce the cross-modal Transformer to fuse the features from different modalities and jointly encode the instructions and the trajectory. The cross-modal Transformer is an 8-layer Transformer encoder [30] with mask. We use eight self-attention heads and a hidden size of 256.

In the teacher-forcing training process, we add a mask when calculating the multi-head self-attention across different modalities. By masking out all the future views in the ground-truth trajectory, the current view $\boldsymbol{v}_t$ is only allowed to refer to the full instructions and all the previous views that the agent has passed by, which is $[\boldsymbol{h}_1^s, \boldsymbol{h}_2^s, \ldots, \boldsymbol{h}_M^s; \boldsymbol{h}_1^v, \boldsymbol{h}_2^v, \ldots, \boldsymbol{h}_{t-1}^v]$, where $M$ denotes the maximum sentence number.

Since the Transformer architecture is based solely on attention mechanism and thus contains no recurrence or convolution, we need to inject additional information about the relative or absolute position of the features in the input

sequence. We add a learned segment embedding to every input feature vector specifying whether it belongs to the sentence encodings or the view encodings. We also add a learned position embedding to indicate the relative position of the sentences in the instruction sequence or the trajectory sequence's views.

the segment encoding, and the position encoding.

**Training Objective** To predict the action $a_t$ for view $\boldsymbol{v}_t$, we concatenate the cross-modal encoder's output of all views in the trajectory up to the current timestamp $t$, and apply a fully-connected layer on top of it.

$$a_t = argmax(\mathcal{FC}(\mathcal{T}(\boldsymbol{h}_1^s||\boldsymbol{h}_2^s||\ldots||\boldsymbol{h}_M^s||\boldsymbol{h}_1^v||\boldsymbol{h}_2^v||\ldots||\boldsymbol{h}_t^v))) \tag{6}$$

where $\mathcal{FC}$ is a fully-connected layer, and $\mathcal{T}$ refers to the Transformer operation.

During training, we use the cross-entropy loss for optimization.

### 3.5 Leverage Multimodal Features

In this section, we discussed our approaches to leveraging the information from different modalities and assisting the VLN task in real-life urban environments. Our MTST framework mainly makes use of the multimodal features in the outdoor navigation datasets in the following three ways:

- We use the trajectory and the masked instruction skeleton in the Touchdown dataset to train our multimodal text style transfer model (MTST). Regarding both the visual features in the trajectory and the textual features in the incomplete instruction template, the MTST model learns to recover the incomplete instruction by injecting object-related information to the generated instruction.
- With the trajectory and masked instruction pairs in the StreetLearn dataset as inference input, we use the MTST model trained on the Touchdown dataset to inference style-modified instructions for StreetLearn trajectories. Such an approach narrows the gap between the instruction styles of the two outdoor navigation datasets.
- We maneuver the StreetLearn trajectories and the style-modified instructions to pre-train our cross-modal reasoning navigator (CMRN). The CMRN learns to fuse and reason through the navigation environment's visual features and the textual features in the instruction. We then fine-tune the CMRN model with the multimodal features in the Touchdown dataset.

## 4 Experiments

### 4.1 Experimental Setup

**Touchdown Dataset** The Touchdown dataset [5, 24] is a dataset for navigation in realistic urban environments. Based on Google Street View[8], Touchdown's navigation environment encompasses 29,641 Street View panoramas of

[8] https://developers.google.com/maps/documentation/streetview/intro

the Manhattan area in New York City, which are connected by 61,319 undirected edges. The dataset contains 9,326 trajectories for the navigation task, and each trajectory is paired with a human-written instruction. The training set consists of 6,526 samples, while the development set and the test set are made up of 1,391 and 1,409 samples, respectively.

**StreetLearn Dataset** The StreetLearn dataset [25] is another dataset for navigation in real-life urban environments based on Google Street View. StreetLearn contains 114k panoramas from the New York City and Pittsburgh. In the StreetLearn navigation environment, the graph for New York City contains 56k nodes and 115k edges, while the graph for Pittsburgh contains 57k nodes and 118k edges. The StreetLearn dataset contains 580k samples in the Manhattan area and 8k samples in the Pittsburgh area for navigation.

While the StreetLearn dataset's trajectory contains more panorama along the way on average, the paired instructions are shorter in length compared to the Touchdown dataset. We extract a sub-dataset *Manh-50* from the original large scale StreetLearn dataset for the convenience of conducting experiments. *Manh-50* consists of navigation samples in the Manhattan area that contains no more than 50 panoramas in the whole trajectory, containing 31k training samples. More statistical details of the dataset can be found in the appendix.

**Dataset Comparison** Even though the Touchdown dataset and the StreetLearn dataset are both built upon Google Street View[9], and both of them contains urban environments in the New York City, pre-training the model with VLN task on the StreetLearn dataset does not raise a threat of test data leaking. This is due to several causes:

- The instructions in the two datasets are distinct in styles. The instructions in the StreetLearn dataset is generated by Google Maps API, which is template-based and focuses on street names. However, the instructions in the Touchdown dataset are created by human annotators and emphasize the visual environment's attributes as navigational cues.
- As reported by [24], the panoramas in the two datasets have little overlaps. In addition, Touchdown instructions constantly refer to transient objects such as cars and bikes, which might not appear in a panorama from a different time. The different granularity of the panorama spacing also leads to distinct panorama distributions of the two datasets.

**Instruction Style Transfer** Among the 9,326 trajectories in the Touchdown dataset, 9,000 are used to train the multimodal text style transfer model, while the rest formed the validation set. We generate style-transferred instructions for the *Manh-50* dataset, which will be used to pre-train the cross-modal reasoning navigator.

[9] https://developers.google.com/maps/documentation/streetview/intro

**Evaluation Metrics** We use the following metrics to evaluate VLN performance:

- Task Completion (TC): the accuracy of completing the navigation task correctly. Following Chen et al. [5], the navigation result is considered correct if the agent reaches the specific goal or one of the adjacent nodes in the environment graph.
- Shortest-Path Distance (SPD): the mean distance between the agent's final position and the goal position in the environment graph.
- Success weighted by Edit Distance (SED): the normalized Levenshtein edit distance between the path predicted by the agent and the reference path, which is constrained only to the successful navigation.
- Coverage weighted by Length Score (CLS): a measurement of the fidelity of the agent's path with respect to reference path.
- Normalized Dynamic Time Warping (nDTW): the minimized cumulative distance between the predicted path and the reference path, normalized by the length of the reference path.
- Success weighted Dynamic Time Warping (SDTW): the nDTW value where the summation is only over the successful navigation.

TC, SPD and SED are defined by Chen et al. [5], CLS is defined by Jain et al. [17], while nDTW and SDTW are defined by Ilharco et al. [15].

**Implementation Details** We pre-train the model with the VLN task on *Manh-50*, the sub-dataset extracted from the StreetLearn dataset. Then, we fine-tune the pre-trained model on the Touchdown dataset for the VLN task.

In the pre-training phase, we use a learning rate of $2.5\times10^{-4}$ for the proposed model. We fine-tune BERT with a learning rate of $1 \times 10^{-5}$. When pre-training on *Manh-50*, the batch size is 30, and the total pre-training epochs are 25.

When training or fine-tuning the model on the Touchdown dataset, the batch size is 36. The learning rate to fine-tune BERT initially set to $1\times10^{-5}$, while the learning rate for other parameters in the model is initialized to be $2.5 \times 10^{-4}$. Adam optimizer [19] is used to optimize all the parameters. More details can be found in the appendix.

### 4.2 Results and Analysis

**Baseline Model** We compare our model with the RCONCAT [5, 25] as the baseline model. The RCONCAT model encodes the trajectory and the instruction in an LSTM-based manner and uses supervised training with Hogwild! [26].

**Quantitative Results** We compare the navigation performance of our CMRN model to the baseline RCONCAT model. We also conduct experiments to compare both models' outdoor navigation results with and without pre-training on external resources. Table 1 presents the navigation results on the Touchdown validation set and test set. We have the following observations from the evaluation results:

| Model | Dev Set | | | | | | Test Set | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TC ↑ | SPD ↓ | SED ↑ | CLS ↑ | nDTW ↑ | SDTW ↑ | TC ↑ | SPD ↓ | SED ↑ | CLS ↑ | nDTW ↑ | SDTW ↑ |
| RCONCAT | 10.9 | 20.2 | 10.5 | 47.8 | 39.2 | 10.6 | 11.7 | 20.8 | 10.8 | 47.2 | 38.5 | 10.8 |
| +*M-50* | 11.4 | 20.5 | 11.0 | 47.9 | 39.7 | 11.0 | 13.5 | 19.6 | 13.1 | 48.9 | 40.9 | 13.2 |
| +*M-50* +style | 11.5 | **19.9** | 11.2 | 48.9 | 40.6 | 11.3 | 13.8 | **18.9** | 13.4 | 50.1 | 42.6 | 13.5 |
| CMRN | 12.9 | 20.2 | 12.5 | 47.1 | 38.9 | 12.5 | 13.1 | 20.1 | 12.8 | 47.4 | 39.6 | 12.9 |
| +*M-50* | 14.4 | 23.7 | 14.0 | 43.4 | 36.3 | 14.0 | 14.3 | 24.7 | 13.9 | 42.1 | 35.0 | 13.9 |
| +*M-50* +style | **15.0** | 20.4 | **14.7** | **49.9** | **42.2** | **14.8** | **16.0** | 21.0 | **15.4** | **50.2** | **42.9** | **15.5** |

Table 1: Quantitative results for the CMRN and the RCONCAT model on outdoor navigation task. +*M-50* denotes pre-training with vanilla *Manh-50* which contains machine-generated instructions; in the +*style* setting, the model is pre-trained with *Manh-50* trajectories and style-modified instructions that are generated by our MTST model.

Firstly, when the navigation model is trained solely on the Touchdown dataset, our CMRN model surpassed the RCONCAT models in all metrics on the test set.

Experimental results show that pre-training on external resources helps improve the task completion rate, and it also improves the navigation performance on the metrics that are calculated on the success cases, such as SED and SDTW. However, pre-training with instructions that have different styles will significantly harm the fidelity of paths generated by our cross-modal reasoning navigator, resulting in a performance drop on SPD, CLS, and nDTW. These results suggest that the difference between the instruction style might misguide the agent in the pre-training stage, and cause the agent to take longer paths in failure cases.

On the other hand, pre-training with style-modified instructions can stably improve navigation performance on all the metrics for the RCONCAT model. It also improves navigation performance on most of the metrics for the CMRN, which means our multimodal text style transfer approach is model-agnostic. The CMRN's inferior performance on SPD indicates that the instructions generated by the MTST model still have certain flaws compared to human-annotated instructions, which might be enhanced in the future study.

### 4.3 Ablation Study

In the ablation studies, we use the following annotations when displaying the evaluation scores: +*M-50* stands for pre-training with vanilla *Manh-50*; in the +*speaker* setting, the instructions are generated by the original Speaker [9], which only attends to the visual input; +*text_attn* denotes that we add a textual attention module to the Speaker so that it can attend to both the visual input and the instruction automatically obtained using Google Map API; in the +*style* setting, the instructions are generated by our MTST model with the "mask-and-recover" learning objective.

**Quality of the Generated Instruction** In the first ablation study, we evaluate the quality of instructions generated by the original Speaker and by the

| Model | BLEU | METEOR | ROUGE_L | CIDEr | SPICE |
|---|---|---|---|---|---|
| +speaker | 15.1 | 20.6 | 22.2 | 1.4 | 20.7 |
| +text_attn | 23.8 | 23.3 | 29.6 | 10.0 | 24.6 |
| +style | **30.5** | **28.8** | **39.7** | **27.8** | **30.7** |

Table 2: Quantitative results of the quality of the instructions generated by the the original Speaker and the MTST model.

| Model | Dev Set | | | | | | Test Set | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TC ↑ | SPD ↓ | SED ↑ | CLS ↑ | nDTW ↑ | SDTW ↑ | TC ↑ | SPD ↓ | SED ↑ | CLS ↑ | nDTW ↑ | SDTW ↑ |
| CMRN +*M-50* | 14.4 | 23.7 | 14.0 | 43.4 | 36.3 | 14.0 | 14.3 | 24.7 | 13.9 | 42.1 | 35.0 | 13.9 |
| +speaker | 7.6 | 26.2 | 7.3 | 34.6 | 26.5 | 7.4 | 8.3 | 25.4 | 8.0 | 36.3 | 27.7 | 8.1 |
| +text_attn | 11.7 | **20.1** | 11.3 | 46.3 | 38.3 | 11.4 | 11.8 | **20.5** | 11.5 | 47.3 | 38.5 | 11.6 |
| +style | **15.0** | 20.4 | **14.7** | **49.9** | **42.2** | **14.8** | **16.0** | 21.0 | **15.4** | **50.2** | **42.9** | **15.5** |

Table 3: Ablation study on the effect of the components in multimodal text style transfer model to the VLN task.

MTST model. We utilize five automatic metrics for natural language generation to evaluate the quality of the generated instructions, including BLEU [27], ROUGE [21], METEOR [8], CIDEr [31] and SPICE [1]. Among the 9,326 trajectories in the Touchdown dataset, 9,000 are used to train the MTST model, while the rest form the validation set.

We report the quantitative results on the validation set for text style transfer in Table 2. After adding textual attention to the original Speaker, the evaluation performance on all five metrics improved. Our MTST model score the highest on all five metrics, which indicates that the "masking-and-recovering" scheme is beneficial for the multimodal text style transfer process and that the MTST model can generate higher quality instructions.

**Multimodal Instruction Style Transfer** We conduct another group of ablation studies to reveal each component's effect in the multimodal instruction style transfer model. Results are shown in Table 3.

According to the evaluation results, the instructions generated by the original Speaker model misguide the navigation agent, which indicates that solely leveraging the Speaker model is not able to reduce the gap between different instruction styles. Adding textual attention to the Speaker model slightly improves the navigation results, but still hinders the agent from navigating correctly. The style-modified instructions improve the agent's performance on all the navigation metrics, which suggests that our multimodal instruction style transfer model can assist the outdoor VLN task.

**Case Study** We demonstrate case study results to illustrate better the performance of our multimodal text style transfer framework. Fig. 5 provides two showcases of the instruction generation results. As listed in the charts, the instructions generated by the original Speaker model not only have a poor performance in keeping the guiding signals in the ground truth instructions but also

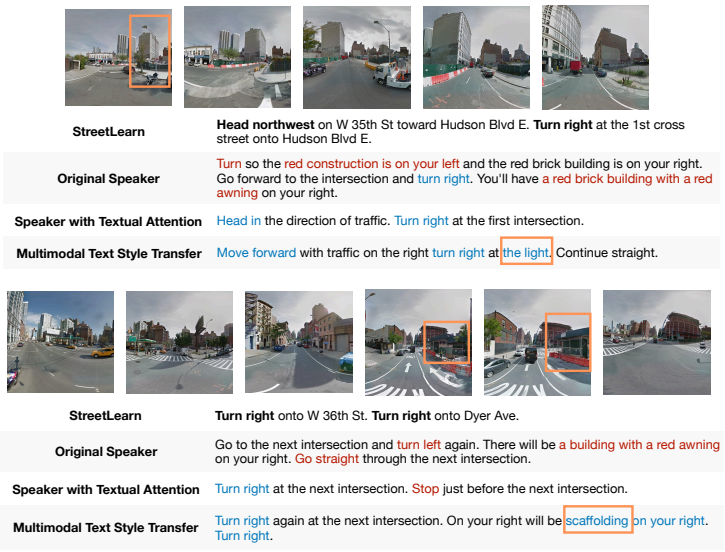



Fig. 5: Two showcases of the instruction generation results. Tokens marked in red indicate having contradictions with the ground truth instruction or the visual trajectory, while the blue tokens suggest alignments. The orange bounding boxes show that the objects in the surrounding environment have been successfully injected into the style-modified instruction.

suffer from hallucinations, which is referring to objects that have not appeared in the trajectory.

The Speaker with textual attention can provide guidance direction. However, the instructions generated in this manner does not utilize the rich visual information in the trajectory. On the other hand, the instructions generated by our multimodal text style transfer model inject more object-related information ("the light", "scaffolding") in the surrounding navigation environment to the StreetLearn instruction, while keeping the correct guiding signals.

## 5 Conclusion

In this paper, we proposed the multimodal text style transfer framework for vision-and-language navigation in real-life urban environments. This learning framework allows us to utilize out-of-domain navigation samples in outdoor environments and enrich the original navigation reasoning training process. Experimental results show that our MTST approach is model-agnostic, and our MTST framework significantly outperforms the baseline models on the outdoor VLN task, improving task completion rate by 22% relatively and achieving new state-of-the-art performance.

# References


[1] P. Anderson, B. Fernando, M. Johnson, and S. Gould. Spice: Semantic propositional image caption evaluation. In *ECCV*, 2016.

[2] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. van den Hengel. Vision-and-Language Navigation: Interpreting Visually-Grounded Navigation Instructions in Real Environments. In *CVPR*, 2018.

[3] S. Bird, E. Klein, and E. Loper. *Natural Language Processing with Python*. O'Reilly Media, 2009.

[4] D. Chen, A. Fisch, J. Weston, and A. Bordes. Reading Wikipedia to Answer Open-Domain Questions. In *ACL*, 2017.

[5] H. Chen, A. Suhr, D. Misra, N. Snavely, and Y. Artzi. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12538–12547, 2019.

[6] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le. AutoAugment: Learning Augmentation Policies from Data. In *CVPR*, 2019.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[8] D. Elliott and F. Keller. Image description using visual dependency representations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1292–1302, 2013.

[9] D. Fried, R. Hu, V. Cirik, A. Rohrbach, J. Andreas, L.-P. Morency, T. Berg-Kirkpatrick, K. Saenko, D. Klein, and T. Darrell. Speaker-follower models for vision-and-language navigation, 2018.

[10] T.-J. Fu, X. Wang, M. Peterson, S. Grafton, M. Eckstein, and W. Y. Wang. Counterfactual Vision-and-Language Navigation via Adversarial Path Sampling. In *arXiv:1911.07308*, 2019.

[11] L. A. Gatys, A. S. Ecker, and M. Bethge. A Neural Algorithm of Artistic Style. In *arxiv:1508.06576*, 2015.

[12] W. Hao, C. Li, X. Li, L. Carin, and J. Gao. Towards Learning a Generic Agent for Vision-and-Language Navigation via Pre-training. In *CVPR*, 2020.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[14] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[15] G. Ilharco, V. Jain, A. Ku, E. Ie, and J. Baldridge. General evaluation for instruction conditioned navigation using dynamic time warping, 2019.

[16] P. T. Jackson, A. Atapour-Abarghouei, S. Bonner, T. Breckon, and B. Obara. Style Augmentation: Data Augmentation via Style Randomization. In *arxiv:1809.05375*, 2018.

[17] V. Jain, G. Magalhaes, A. Ku, A. Vaswani, E. Ie, and J. Baldridge. Stay on the path: Instruction fidelity in vision-and-language navigation. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. https://doi.org/10.18653/v1/p19-1181. URL `http://dx.doi.org/10.18653/V1/P19-1181`.

[18] L. Ke, X. Li1, Y. Bisk, A. Holtzman, Z. Gan, J. Liu, J. Gao, Y. Choi, and S. Srinivasa. Tactical Rewind: Self-Correction via Backtracking in Vision-and-Language Navigation. In *CVPR*, 2019.

[19] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2014.

[20] L. Lansing, V. Jain, H. Mehta, H. Huang, and E. Ie. Valan: Vision and language agent navigation, 2019.

[21] C.-Y. Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/W04-1013`.

[22] C.-Y. Ma, J. Lu, Z. Wu, G. AlRegib, Z. Kira, R. Socher, and C. Xiong. Self-monitoring navigation agent via auxiliary progress estimation. *arXiv preprint arXiv:1901.03035*, 2019.

[23] C.-Y. Ma, Z. Wu, G. AlRegib, C. Xiong, and Z. Kira. The Regretful Agent: Heuristic-Aided Navigation through Progress Estimation. In *CVPR*, 2019.

[24] H. Mehta, Y. Artzi, J. Baldridge, E. Ie, and P. Mirowski. Retouchdown: Adding touchdown to streetlearn as a shareable resource for language grounding tasks in street view, 2020.

[25] P. Mirowski, M. K. Grimes, M. Malinowski, K. M. Hermann, K. Anderson, D. Teplyashin, K. Simonyan, K. Kavukcuoglu, A. Zisserman, and R. Hadsell. Learning to navigate in cities without a map, 2018.

[26] F. Niu, B. Recht, C. Re, and S. J. Wright. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent, 2011.

[27] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[28] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge, 2014.

[29] H. Tan, L. Yu, and M. Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. *arXiv preprint arXiv:1904.04195*, 2019.

[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2017.

[31] R. Vedantam, C. Lawrence Zitnick, and D. Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575, 2015.

[32] X. Wang, W. Xiong, H. Wang, and W. Y. Wang. Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. *Lecture Notes in Computer Science*,

page 38–55, 2018. ISSN 1611-3349. https://doi.org/10.1007/978-3-030-01270-0-3. URL http://dx.doi.org/10.1007/978-3-030-01270-0-3.

[33] X. Wang, Q. Huang, A. Celikyilmaz, J. Gao, D. Shen, Y.-F. Wang, W. Y. Wang, and L. Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6629–6638, 2019.

[34] X. Wang, J. Wu, D. Zhang, Y. Su, and W. Y. Wang. Learning to Compose Topic-Aware Mixture of Experts for Zero-Shot Video Captioning. In *AAAI*, 2019.

[35] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

[36] Y. Xu and A. Goel. Cross-Domain Image Classification through Neural-Style Transfer Data Augmentation. In *arxiv:1910.05611*, 2019.

[37] Z. Zheng, S. Huang, Z. Sun, R. Weng, X.-Y. Dai, and J. Chen. Learning to Discriminate Noises for Incorporating External Information in Neural Machine Translation. In *arxiv:1810.10317*, 2018.

# A Appendix

## A.1 Dataset Comparison

Table 4 lists out the statistical information of the datasets used in pre-training and fine-tuning. We can see that the StreetLearn dataset has longer trajectories than the Touchdown dataset, which its instructions are significantly shorter in length.

| Dataset | #data | #pano | instr_len | #sent | #turn | #covered |
|---|---|---|---|---|---|---|
| Touchdown | 6k | 35.2 | 80.5 | 6.3 | 2.8 | 26k |
| Manh-50 | 31k | 37.2 | 22.1 | 2.8 | 4.1 | 43k |
| StreetLearn | 580k | 129.0 | 28.6 | 4.0 | 13.2 | 114k |

Table 4: Statistical information of the datasets used in pre-training and fine-tuning. *#data* refers to the total lines of samples in the training set, *#pano* indicates how many panoramas each trajectory contains on average, *instr_len* is the average length of the instructions, *#sent* denotes how many sentences each instruction contains, *#turn* points to the average number of panoramas that stand as intersections in each trajectory, *#covered* is the total number of panoramas that are covered by the training set.

## A.2 Instruction Style

The instructions in the StreetLearn dataset [25], a large-scale interactive navigation environment built upon Google Street View[10], is generated by Google Maps API. Street names are always mentioned in the templated-based instructions in StreetLearn. As a result, these template-based instructions always mention street names when providing suggestions for navigation actions. However, static information such as street names is not directly revealed in the navigation environment, mainly when the agent does not acquire a top-down view of the overall environment.

The instructions in the Touchdown dataset [5, 24], another urban navigation environment with Street View panoramas, is written by human annotators. These natural language instructions frequently refer to objects in the panorama, such as traffic lights, cars, awnings, etc.

Table 5 lists out two instruction samples in the StreetLearn dataset and the Touchdown dataset.

## A.3 View Encoder Implementation

Following Chen et al. [5], we embed each panorama $\mathbf{I}_t$ by slicing it into eight images and projecting each image from an equirectangular projection to a perspective projection. Each of the projected image of size $800 \times 460$ will be passed

[10] https://developers.google.com/maps/documentation/streetview/intro

| Dataset | Instruction |
|---|---|
| StreetLearn | Head northwest on E 23rd St toward 2nd Ave. Turn left at the 2nd cross street onto 3rd Ave. Turn right at the 2nd cross street onto E 21st St. |
| Touchdown | Orient yourself so you are facing the same as the traffic on the 4 lane road. Travel down this road until the first intersection. Turn left and go down this street with the flow of traffic. You'll see a black and white stripped awning on your right as you travel down the street. Keep going pass the parking building on your right, until you are right next to a large open red dumpster. |

Table 5: Instructions in the StreetLearn dataset and the Touchdown dataset are different in style. The templated-based StreetLearn instructions are machine-generated and heavily rely on street information, such as E 23rd St, 2nd Ave, and 3rd Ave. In contrast, the human-annotated instructions in Touchdown pay more attention to the surrounding objects, such as 4 lane road, black and white stripped awning, and large open red dumpster.

through the RESNET18 [13] pre-trained on ImageNet [28]. We use the output of size $128 \times 100 \times 58$ from the fourth to last layer before classification as the feature for each slice. The feature map for each panorama is the concatenation of the eight image slices, which is a single tensor of size $128 \times 100 \times 464$.

We center the feature map according to the agent's heading $\alpha_t$ at time stamp $t$. We crop a $128 \times 100 \times 100$ sized feature map from the center and calculate the mean value along the channel dimension. The resulting $100 \times 100$ features is regard as the current panorama feature $\hat{\mathbf{I}}_t$ for each state. Following Mirowski et al. [25], we then apply a three-layer convolutional neural network on $\hat{\mathbf{I}}_t$ to extract the view features $\boldsymbol{h}_t^v \in \mathbb{R}^{256}$ at time stamp $t$.

The first layer has one input channel and 32 output channels, using $8 \times 8$ kernels with stride 4. The second layer has 32 input channels and 64 output channels, using $4 \times 4$ kernels with stride 4. ReLu is applied as the activation function after each convolutional operation. The convolutional layer's output is projected by a single fully-connected layer to receive the view feature representation $\boldsymbol{h}_t^v \in \mathbb{R}^{256}$.