Analyzing Generalization of Vision and Language Navigation to Unseen Outdoor Areas

Raphael Schumann
Computational Linguistics
Heidelberg University, Germany

Stefan Riezler
Computational Linguistics & IWR
Heidelberg University, Germany

{rschuman|riezler}@cl.uni-heidelberg.de

Abstract

Vision and language navigation (VLN) is a challenging visually-grounded language understanding task. Given a natural language navigation instruction, a visual agent interacts with a graph-based environment equipped with panorama images and tries to follow the described route. Most prior work has been conducted in indoor scenarios where best results were obtained for navigation on routes that are similar to the training routes, with sharp drops in performance when testing on unseen environments. We focus on VLN in outdoor scenarios and find that in contrast to indoor VLN, most of the gain in outdoor VLN on unseen data is due to features like junction type embedding or heading delta that are specific to the respective environment graph, while image information plays a very minor role in generalizing VLN to unseen outdoor areas. These findings show a bias to specifics of graph representations of urban environments, demanding that VLN tasks grow in scale and diversity of geographical environments.[1]

1 Introduction

Vision and language navigation (VLN) is a challenging task that requires the agent to process natural language instructions and ground them in a visual environment. The agent is embodied in the environment and receives navigation instructions. Based on the instructions, the observed surroundings, and the current trajectory the agent decides its next action. Executing this action changes the position and/or heading of the agent within the environment, and eventually the agent follows the described route and stops at the desired goal location. The most common evaluation metric in VLN is the proportion of successful agent navigations, called task completion (TC).

While early work on grounded navigation was confined to grid-world scenarios (MacMahon et al., 2006; Chen and Mooney, 2011), recent work has studied VLN in outdoor environment consisting of real-world urban street layouts and corresponding panorama pictures (Chen et al., 2019). Recent agent models for outdoor VLN treat the task as a sequence-to-sequence problem where the instructions text is the input and the output is a sequence of actions (Chen et al., 2019; Xiang et al., 2020; Zhu et al., 2021b). In contrast to indoor VLN (Anderson et al., 2018; Ku et al., 2020), these works only consider a *seen scenario*, i.e., the agent is tested on routes that are located in the same area as the training routes. However, studies of indoor VLN (Zhang et al., 2020) show a significant performance drop when testing in previously unseen areas.

The main goal of our work is to study *outdoor VLN in unseen areas*, pursuing the research question of which representations of an environment and of instructions an agent needs to succeed at this task. We compare existing approaches to a new approach that utilizes features based on the observed environment graph to improve generalization to unseen areas. The first feature, called junction type embedding, encodes the number of outgoing edges at the current agent position; the second feature, called heading delta, encodes the agent's heading change relative to the previous timestep. As our experimental studies show, representations of full images do not contribute very much to successful VLN in outdoor scenarios beyond these two features. One reason why restricted features encoding junction type and heading delta are successful in this task is that they seem to be sufficient to encode peculiarities of the graph representation of the environments. Another reason is the current restriction of outdoor environments to small urban areas. In our case, one dataset is the widely used Touchdown dataset introduced by Chen et al. (2019), the

[1] Code: https://github.com/raphael-sch/map2seq_vln
Data & Demo: https://map2seq.schumann.pub/vln/

other dataset is called map2seq and has recently been introduced by Schumann and Riezler (2021). The map2seq dataset was created for the task of navigation instructions generation but can directly be adopted to VLN. We conduct a detailed analysis of the influence of general neural architectures, specific features such as junction type or heading delta, the role of image information and instruction token types, to outdoor VLN in seen and unseen environments on these two datasets.

Our specific findings unravel the contributions of these features on several VLN subtasks such as orientation, directions, stopping. Our general finding is that current outdoor VLN suffers a bias towards urban environments and to artifacts of their graph representation, showing the necessity of more diverse datasets and tasks for outdoor VLN.

Our main contributions are the following:

- We describe a straightforward agent model that achieves state-of-the-art task completion and is used as a basis for our experiments.
- We introduce the *unseen scenario* for outdoor VLN and propose two environment-dependent features to improve generalization in that setting.
- We compare different visual representations and conduct language masking experiments to study the effect in the unseen scenario.
- We adopt the map2seq dataset to VLN and show that merging it with Touchdown improves performance on the respective test sets.

2 VLN Problem Definition

The goal of the agent is to follow a route and stop at the desired target location based on natural language navigation instructions. The environment is a directed graph with nodes $v \in \mathbb{V}$ and labeled edges $(u, v) \in \mathbb{E}$. Each node is associated with a 360° panorama image p and each edge is labeled with an angle $\alpha_{(u,v)}$. The agent state $s \in S$ consists of a node and the angle at which the agent is heading: $(v, \alpha_{(v,u)} \mid u \in \mathbb{N}_v^{out})$, where $\mathbb{N}_v^{out}$ are all outgoing neighbors of node v. The agent can navigate the environment by performing an action $a \in \{\texttt{FORWARD}, \texttt{LEFT}, \texttt{RIGHT}, \texttt{STOP}\}$ at each timestep t. The `FORWARD` action moves the agent from state $(v, \alpha_{(v,u)})$ to $(u, \alpha_{(u,u')})$, where (u, u') is the edge with an angle closest to $\alpha_{(v,u)}$. The `RIGHT` and `LEFT` action rotates the agent towards the closest edge angle in clockwise or counterclockwise direction, respectively: $(v, \alpha_{(v,u')})$. Given a starting state s_1 and instructions text $\mathbf{x}$, the agent performs a series of actions $a_1, ..., a_T$ until the `STOP` action is predicted. If the agent stops within one neighboring node of the desired target node (goal location), the navigation was successful. The described environment and location finding task was first introduced by (Chen et al., 2019) and we will also refer to it as "outdoor VLN task" throughout this paper.

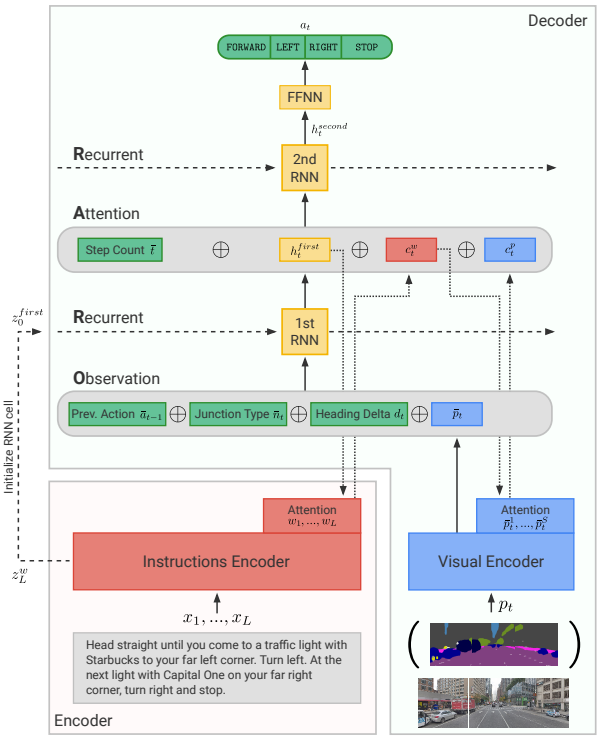

Figure 1: The ORAR model for outdoor vision and language navigation follows a sequence-to-sequence architecture. The instructions text is encoded and used along the visual features to predict the next agent action. The recurrent decoder has two layers, the first encodes observations about the current environment state, the second allows attention over the input text and panorama view. The predicted action changes the state of the agent in the environment and with it the panorama view of the next timestep.

3 Model Architecture

In this section we introduce the model that we use to analyze navigation performance in the unseen and seen scenario for outdoor VLN. The architecture is inspired by the cross-modal attention model for indoor VLN (Krantz et al., 2020). First we give a high level overview of the model architecture and rough intuition. Afterwards we provide a more

formal description.

As depicted in Figure 1, the model follows a sequence-to-sequence architecture where the input sequence is the navigation instructions text and the output is a sequence of agent actions. At each decoding timestep, a new visual representation of the current agent state within the environment is computed, where the agent state is dependent on the previously predicted actions. The decoder RNN has two layers where the first encodes metadata and a visual representation. The second RNN layer encodes a contextualized text and visual representation and eventually predicts the next action.

The intuition behind the model architecture is to firstly accumulate plain *observations* available at the current timestep and entangle them with previous observations in the first *recurrent* layer. Based on these observations, the model focuses *attention* to certain parts of the instructions text and visual features which are again entangled in the second *recurrent* layer. Thus, we use the acronym *ORAR* (observation-recurrence attention-recurrence) for the model.

In detail, the instructions encoder embeds and encodes the tokens in the navigation instructions sequence $\mathbf{x} = x_1, ..., x_L$ using a bidirectional LSTM (Graves et al., 2005):

$$\hat{x}_i = \text{embedding}(x_i)$$
$$((w_1, ..., w_L), z_L^w) = \text{Bi-LSTM}(\hat{x}_1, ..., \hat{x}_L),$$

where $w_1, ..., w_L$ are the hidden representations for each token and z_L^w is the last LSTM cell state. The visual encoder, described in detail below, emits a fixed size representation $\bar{p}_t$ of the current panorama view and a sequence of sliced view representations $\bar{p}_t^1, ..., \bar{p}_t^S$. The state z_0^{first} of the cell in the first decoder LSTM layer is initialized using z_L^w. The input to the first decoder layer is the concatenation ($\oplus$) of visual representation $\bar{p}_t$, previous action embedding $\bar{a}_{t-1}$, junction type embedding $\bar{n}_t$, and heading delta d_t. The output of the first decoder layer,

$$h_t^{first} = \text{LSTM}^{first}([\bar{a}_{t-1} \oplus \bar{n}_t \oplus d_t \oplus \bar{p}_t]),$$

is then used as the query of multi-head attention (Vaswani et al., 2017) over the text encoder. The resulting contextualized text representation c_t^w is then used to attend over the sliced visual representations:

$$c_t^w = \text{MultiHeadAttention}(h_t^{first}, (w_1, ..., w_L))$$
$$c_t^p = \text{MultiHeadAttention}(c_t^w, (\bar{p}_t^1, ..., \bar{p}_t^S)).$$

The input and output of the second decoder layer are

$$h_t^{second} = \text{LSTM}^{second}([\bar{t} \oplus h_t^{first} \oplus c_t^w \oplus c_t^p]),$$

where $\bar{t}$ is the embedded timestep t. The hidden representation h_t^{second} of the second decoder LSTM layer is then passed through a feed forward network to predict the next agent action a_t.

3.1 Visual Encoder

At each timestep t the panorama at the current agent position is represented by extracted visual features. We slice the panorama into eight projected rectangles with 60° field of view, such that one of the slices aligns with the agent's heading. This centering slice and the two left and right of it are fed into a ResNet pretrained[2] on ImageNet (Russakovsky et al., 2015). We consider two variants of ResNet derived panorama features. One variant extracts low level features from the fourth to last layer (**4th-to-last**) of a pretrained ResNet-18 and concatenates each slice's feature map along the width dimension, averages the 128 CNN filters and cuts out 100 dimensions around the agents heading. This results in a feature matrix of 100×100 ($\bar{p}_t^1, ..., \bar{p}_t^{100}$). The full procedure is described in detail in Chen et al. (2019) and Zhu et al. (2021b). The other variant extracts high level features from a pretrained ResNet-50's **pre-final** layer for each of the 5 slices: $\bar{p}_t^1, ..., \bar{p}_t^5$. Each slice vector $\bar{p}_t^s$ is of size $2,048$ resulting in roughly the same number of extracted ResNet features for both variants, making a fair comparison. Further, we use the **semantic segmentation** representation of the panorama images. We employ omnidirectional semantic segmentation (Yang et al., 2020) to classify each pixel by one of the 25 classes of the Mapillary Vistas dataset (Neuhold et al., 2017). The classes include e.g. car, truck, traffic light, vegetation, road, sidewalk. See Figure 1 bottom right for a visualization. Each panorama slice ($\bar{p}_t^1, ..., \bar{p}_t^5$) is then represented by a 25 dimensional vector where each value is the normalized area covered by the corresponding class (Zhang et al., 2020). For either feature extraction method, the fixed sized panorama representation $\bar{p}_t$ is computed by concatenating the slice features $\bar{p}_t^1, ..., \bar{p}_t^S$ and passing them to a feed forward network.

[2] https://pytorch.org/vision/0.8/models.html

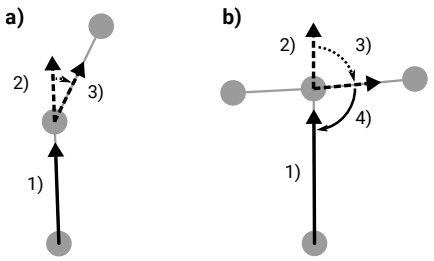

Figure 2: Visualization of automatic agent rotation initiated by the environment. Grey circles and interconnecting edges are part of the environment graph. Black solid arrows are actions initiated by the agent. Black dotted arrows depict agent heading and automatic rotation by the environment. **a)**: 1) The agent moves forward. 2) Agent's heading does not point to an outgoing edge. 3) Agent is automatically rotated to the closest edge without causing problems. **b)**: The agent receives instructions like "Turn right at the next intersection". 1) The agent moves forward. 2) Agent's heading does not point to an outgoing edge. 3) The environment automatically rotates the agent towards the closest outgoing edge. 4) The agent has no explicit information about the automatic rotation and predicts a right turn as instructed, leading to a failed navigation.

3.2 Junction Type Embedding

The junction type embedding is a feature that we introduce to better analyze generalization to unseen areas. It embeds the number of outgoing edges of the current environment node and is categorized into {2, 3, 4, >4}. It provides the agent information about the type of junction it is positioned on: a regular street segment, a three-way intersection, a four way intersection or an intersection with more than four outgoing streets. We want to point out that the number of outgoing edges isn't oracle information in the environment described in Section 2. The agent can rotate left until the same panorama view is observed and thus counting the number of outgoing edges by purely interacting with the environment. But it is clear that the feature leverages the fact that the environment is based on a graph and it would not be available in a continuous setting (Krantz et al., 2020).

3.3 Heading Delta

As described in Section 2, the environment defined and implemented by Chen et al. (2019) only allows states where the agent is heading towards an outgoing edge. As a consequence the environment automatically rotates the agent towards the closest outgoing edge after transitioning to a new node. The environment behavior is depicted in Figure 2a) for a transition between two regular street segments. However, as depicted in Figure 2b), a problem arises when the agent is walking towards a three-way intersection. The automatic rotation introduces unpredictable behavior for the agent and we hypothesis that it hinders generalization to unseen areas. To correct for this environment artifact, we introduce the heading delta feature d_t which encodes the change in heading direction relative to the previous timestep. The feature is normalized to $(-1, 1]$ where a negative value indicates a left rotation and a positive value indicates a right rotation. The magnitude signals the degree of the rotation up to 180°.

4 Data

We use the Touchdown (Chen et al., 2019) and the map2seq (Schumann and Riezler, 2021) datasets in our experiments. Both datasets contain human written navigation instructions for routes located in the same environment. The environment consists of 29,641 panorama images from Manhattan and the corresponding connectivity graph.

4.1 Touchdown

The Touchdown dataset (Chen et al., 2019) for vision and language navigation consists of 9,326 routes paired with human written navigation instructions. The annotators navigated the panorama environment based on a predefined route and wrote down navigation instructions along the way.

4.2 Map2seq

The map2seq (Schumann and Riezler, 2021) dataset was created for the task of navigation instructions generation. The 7,672 navigation instructions were written by human annotators who saw a route on a rendered map, without the corresponding panorama images. The annotators were told to include visual landmarks like stores, parks, churches, and other amenities into their instructions. A different annotator later validated the written navigation instructions by using them to follow the described route in the panorama environment (without the map). This annotation procedure allows us to use the navigation instructions in the map2seq dataset for the vision and language navigation task. We are the first to report VLN results on this dataset.

4.3 Comparison

Despite being located in the same environment, the routes and instructions from each dataset differ in

multiple aspects. The map2seq instructions typically include named entities like store names, while Touchdown instructions focus more on visual features like the color of a store. Both do not include street names or cardinal directions and are written in egocentric perspective. Further, in map2seq the agent starts by facing in the correct direction, while in Touchdown the initial heading is random and the first part of the instruction is about orientating the agent ("Turn around such that the scaffolding is on your right"). A route in map2seq includes a minimum of three intersections and is the shortest path from the start to the end location.[3] In Touchdown there are no such constraints and a route can almost be circular. The routes in both datasets are around 35-45 nodes long with some shorter outliers in Touchdown. On average instructions are around 55 tokens long in map2seq and around 89 tokens long in Touchdown.

5 Experiments

We are interested in the generalization ability to unseen areas and how it is influenced by the two proposed features, types of visual representation, navigation instructions and training set size. Alongside of the results in the unseen scenario, we report results in the seen scenario to interpret performance improvements in relation to each other. All experiments[4] are repeated ten times with different random seeds. The reported numbers are the average over the ten repetitions. Results printed in **bold** are significantly better than non-bold results in the same column. Significance was established by a paired t-test[5] on the ten repetition results and a p-value ≤ 0.05 without multiple hypothesis corrections factor. Individual results can be found in the Appendix.

5.1 Data Splits

To be able to compare our model with previous work, we use the original training, development and test split (Chen et al., 2019) for the seen scenario on Touchdown. Because we are the first to use the map2seq data for VLN we create a new split for it. The resulting number of instances can be seen in the left column of Table 1. For the unseen scenario, we create new splits for both datasets. We separate the unseen area geographically by drawing a boundary across lower Manhattan (see Figure 3). Development and test instances are randomly chosen from within the unseen area. Routes that are crossing the boundary are discarded. The right column of Table 1 shows the number of instances for both splits. Additionally, we merge the two datasets for both scenarios. This is possible because both datasets are located in the same environment and the unseen boundary is equivalent.

Figure 3: Visualization of the environment area located in Manhattan. **The seen scenario is depicted on the left and the unseen scenario on the right.** Each white dot is a training route and each black dot is a test route in the Touchdown and map2seq dataset. The unseen scenario is characterized by geographic separation of the training and testing area.

	seen			unseen		
	train	**dev**	**test**	**train**	**dev**	**test**
Touchdown	6,525	1,391	1,409	6,770	800	1,507
map2seq	6,072	800	800	5,737	800	800
Merged	12,597	2,191	2,209	12,507	1,600	2,307

Table 1: Number of instances in the data splits for the seen and unseen scenario of Touchdown and map2seq.

5.2 Training Details

We train the models with Adam (Kingma and Ba, 2015) by minimizing cross entropy loss in the teacher forcing paradigm. We set the learning rate to 5e-4, weight decay to 1e-3 and batch size to 64. After 150 epochs we select the model with the best shortest path distance (SPD) performance on the development set. We apply dropout of 0.3 after each dense layer and recurrent connection. The multi-head attention mechanism is regularized

[3] The shortest path bias reduces the number of reasonable directions at each intersection and thus makes the task easier.

[4] Except comparison models on the Touchdown seen test set for which we copy the results from the respective work.

[5] https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest_rel.html

	Seen								Unseen							
	Touchdown				map2seq				Touchdown				map2seq			
	dev		test		dev		test		dev		test		dev		test	
Model	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC
RConcat	22.5	10.6	22.9	11.8	30.7	17.1	27.7	14.7	3.9	2.3	3.5	1.9	3.7	2.0	3.8	2.1
GA	25.2	12.0	24.9	11.9	33.0	18.2	30.1	17.0	3.6	1.8	4.0	2.2	3.9	1.8	4.1	1.7
ARC	-	15.3	-	14.1	-	-	-	-	-	-	-	-	-	-	-	-
ARC+l2s	-	19.5	-	16.7	-	-	-	-	-	-	-	-	-	-	-	-
VLN Transformer	23.0	14.0	25.3	14.9	31.1	18.6	29.5	17.0	4.7	2.3	5.2	3.1	6.2	3.6	6.1	3.5
ORAR full model																
• ResNet pre-final	38.9	26.0	38.4	25.3	**65.0**	49.1	**62.3**	**46.7**	13.0	9.6	12.1	8.8	34.6	24.2	34.5	24.6
• ResNet 4th-to-last	**45.1**	**29.9**	**44.9**	**29.1**	60.0	43.4	57.8	41.7	**22.2**	**15.4**	**21.6**	**14.9**	**41.0**	**27.6**	**42.2**	**30.3**
ORAR full model	• ResNet 4th-to-last				• ResNet pre-final				• ResNet 4th-to-last				• ResNet 4th-to-last			
- no heading delta	**45.5**	**30.0**	**45.3**	**29.3**	63.2	47.7	**60.3**	44.9	**21.6**	**15.2**	**21.2**	**14.8**	33.0	22.0	33.6	23.6
- no junction type	40.6	25.9	40.9	25.5	**65.9**	**52.9**	**62.1**	**47.5**	7.9	4.8	7.1	4.3	13.1	7.4	11.8	7.1
- no head. & no junc.	39.2	24.6	39.4	24.2	62.7	**49.6**	58.9	45.1	7.6	4.6	7.0	4.4	8.9	5.0	8.2	4.7

Table 2: Results on Touchdown and map2seq for the seen and unseen scenario. Metrics are normalized Dynamic Time Warping (nDTW) and task completion (TC). In the first section we list results for the comparison models: RConcat, GA, VLN Transformer (Zhu et al., 2021b) and ARC, ARC+learn2stop (Xiang et al., 2020). In the second section we present results for the ORAR model with two different types of image features: *ResNet pre-final* features are extracted from the last layer before the classification and *ResNet 4th-to-last* are low level features extracted from the fourth to last layer of a pretrained ResNet. The last section ablates the two proposed features: *heading delta* and *junction type embedding*.

by attention dropout of 0.3 and layer normalization. The navigation instructions are lower-cased and split into byte pair encodings (Sennrich et al., 2016) with a vocabulary of 2,000 tokens and we use BPE dropout (Provilkov et al., 2020) during training. The BPE embeddings are of size 32 and the bidirectional encoder LSTM has two layers of size 256. The feed forward network in the visual encoder consists of two dense layers with 512 and 256 neurons, respectively, and 64 neurons in case of using semantic segmentation features. The embeddings that encode previous action, junction type, and step count are of size 16. The two decoder LSTM layers are of size 256 and we use two attention heads. Training the full model takes around 3 hours on a GTX 1080 Ti.

5.3 Model Comparison

We compare the ORAR model to previous works. Because these works only report results for the seen scenario on Touchdown, we evaluate those for which we could acquire the code, on the map2seq dataset and the unseen scenario. The models *RConcat* (Mirowski et al., 2018; Chen et al., 2019), *GA* (Chaplot et al., 2018; Chen et al., 2019) and *ARC* (Xiang et al., 2020) use an LSTM to encode the instructions text and a single layer decoder LSTM to predict the next action. They differ in how the text and image representations are incorporated during each timestep in the decoder. As the name suggests, in *RConcat* the two representations are concatenated. *GA* uses gated attention to compute a fused representation of text and image. *ARC* uses the hidden representation of the previous timestep to attend over the instructions text. This contextualized text representation is then concatenated to the image representation. They further introduce *ARC+l2s* which cascades the action prediction into a binary stopping decision and a subsequent direction classification. The *VLN-Transformer* (Zhu et al., 2021b) uses pretrained BERT (Devlin et al., 2019) to encode the instructions and VLN-BERT (Majumdar et al., 2020) to fuse the modalities.

5.4 Metrics

We use task completion (**TC**) as the main performance metric. It represents the percentage of successful agent navigations (Chen et al., 2019). We further report normalized Dynamic Time Warping (**nDTW**) which quantifies agent and gold trajectory overlap for all routes (Ilharco et al., 2019). The shortest path distance (**SPD**) is measured within the environment graph from the node the agent stopped to the goal node (Chen et al., 2019).

6 Results & Analysis

The two upper sections of Table 2 show the results of the ORAR model introduced in Section 3 in comparison to other work. While the model sig-

	Unseen			
	Touchdown		map2seq	
Visual Features	**dev**	**test**	**dev**	**test**
ResNet pre-final	9.6	8.8	24.2	24.6
- no junction type	4.4	4.0	10.7	11.0
ResNet 4th-to-last	**15.4**	**14.9**	27.6	**30.3**
- no junction type	**4.8**	4.3	7.4	7.1
semantic segmentation	11.5	11.0	**29.0**	**31.1**
- no junction type	**5.5**	**5.5**	**11.6**	**12.1**
no image	11.5	9.5	**28.5**	**30.5**
- no junction type	3.0	2.8	5.4	5.5

Table 3: Study of visual features for the unseen scenario of Touchdown and map2seq. Metric is task completion.

nificantly outperforms all previous work on both datasets, our main focus is analyzing generalization to the unseen scenario. It is apparent that the type of image features influences agent performance and will be discussed in the next section. The bottom section of Table 2 ablates the proposed heading delta and junction type features for the best models. Removing the heading delta feature has little impact in the seen scenario, but significantly reduces task completion in the unseen scenario of the map2seq dataset. Surprisingly, the feature has no impact in the unseen scenario of Touchdown. We believe this is a consequence of the different data collection processes. Touchdown was specifically collected for VLN and annotators navigated the environment graph, while map2seq annotators wrote instructions only seeing the map. Removing the junction type embedding leads to a collapse of task completion in the unseen scenario on both datasets. This shows that without this explicit feature, the agent lacks the ability to reliably identify intersections in new areas.

6.1 Visual Features

Table 3 shows results for different types of visual features in the unseen scenario. We compare high level ResNet features (pre-final), low level ResNet features (4th-to-last), semantic segmentation features and using no image features. For the ResNet based features, the low level 4th-to-last features perform better than pre-final on both datasets. On map2seq the no image baseline performs on par with models that have access to visual features. When we remove the junction type embedding, the task completion rate drops significantly, which shows that the agent is not able to reliably locate intersections from any type of visual features.

	Touchdown			
	Seen		Unseen	
Sub-task	**dev**	**test**	**dev**	**test**
ORAR pre-final	26.0	25.3	9.6	8.8
orientation	79.2	77.5	66.7	67.6
directions	84.8	85.5	45.9	45.7
stopping	40.7	41.0	37.4	36.1
ORAR 4th layer	**29.9**	**29.1**	**15.4**	**14.9**
orientation	92.4	91.5	84.2	84.1
directions	81.6	81.1	53.4	52.4
stopping	39.7	40.2	36.4	35.2
ORAR no image	15.2	13.3	11.1	9.5
orientation	59.8	57.0	61.3	60.5
directions	74.1	73.3	58.8	57.9
stopping	39.3	38.8	36.1	34.0

Table 4: Oracle analysis on Touchdown. Division into three sub-tasks: *orientation*, *directions* and *stopping*. Providing oracle actions for two of the three sub-tasks allows an isolated look at the remaining one. Underlined results are best for the sub-task, e.g. 85.5 is the best TC for the directions task on the test set in the seen scenario.

6.2 Sub-task Oracle

The agent has to predict a sequence of actions in order to successfully reach the goal location. In Touchdown this task can be divided into three sub-tasks (see Section 4). First the agent needs to **orientate** itself towards the correct starting heading. Next the agent has to predict the correct **directions** at the intersections along the path. The third sub-task is **stopping** at the specified location. Providing oracle actions (during testing) for two of the three sub-tasks lets us look at the completion rate of the remaining sub-task. Table 4 shows the completion rates for each of the three sub-tasks when using ResNet pre-final, 4th-to-last and no image features. In the seen scenario we can observe that the pre-final features lead to the best performance for the directions task. The 4th-to-last features on the other hand lead to the best orientation task performance and the stopping task is not influenced by the choice of visual features. In the unseen scenario 4th-to-last features again provide best orientation task performance but no image features lead to the best performance for the directions task. This shows that the ResNet 4th-to-last features are primarily useful for the orientation sub-task and explains the discrepancy of the no image baseline on Touchdown and map2seq identified in the previous subsection. In the Appendix we use this knowledge to train a mixed-model that uses 4th-to-last features

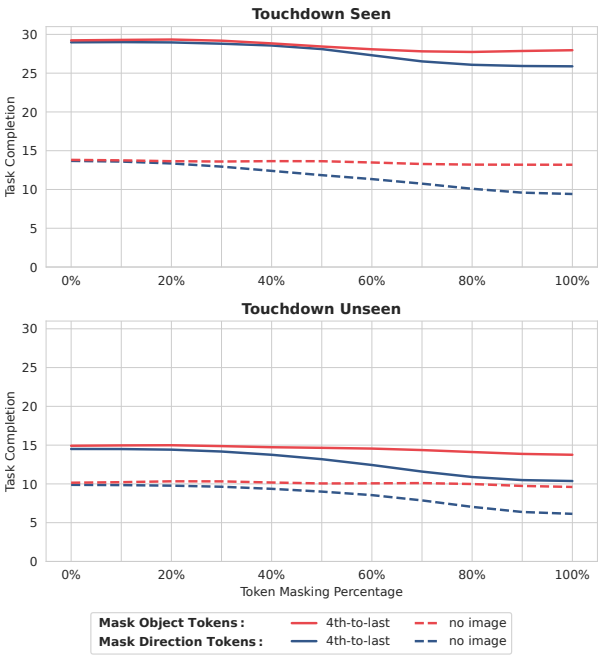

Figure 4: Masking experiments on the seen and unseen test set of Touchdown. Object or direction tokens are masked during training and testing.

for the orientation sub-task and pre-final/no image features for directions and stopping.

6.3 Token Masking

To analyze the importance of direction and object tokens in the navigation instructions, we run masking experiments similar to Zhu et al. (2021a), except that we mask the tokens during training and testing instead of during testing only. Figure 4 shows the resulting task completion rates for an increasing number of masked direction or object tokens. From the widening gap between masking object and direction tokens, we can see that the direction tokens are more important to successfully reach the goal location. Task completion nearly doesn't change when masking object tokens, indicating that they are mostly ignored by the model. While task completion significantly drops when direction tokens are masked, the agent still performs on a high level. This finding is surprising and in dissent with Zhu et al. (2021a) who report that task completion nearly drops to zero when masking direction tokens during testing only. We believe that in our setting (masking during testing and training), the model learns to infer the correct directions from redundancies in the instructions or context around the direction tokens. Besides the general trend of lower performance on the unseen scenario, we can not identify different utilization of object or direction tokens in the seen and unseen scenario.

6.4 Merged Datasets

We train the ORAR full model on the merged dataset (see Section 5.1). Model selection is performed on the merged development set but results are also reported for the individual test sets of Touchdown and map2seq. For comparison with models trained on the non-merged datasets, the first row of Table 5 shows the best results of Table 2. Training on the merged dataset significantly improves nDTW and task completion across both datasets and scenarios. This shows that both datasets are compatible and the merged dataset can further be used by the VLN community to evaluate their models on more diverse navigation instructions. Despite being trained on twice as many instances, the no image baseline still performs on par on map2seq unseen. From this we conclude that the current bottleneck for better generalization to unseen areas is the number of panorama images seen during training instead of number of instructions.

7 Related Work

Natural language instructed navigation of embodied agents has been studied in generated grid environments that allow a structured representation of the observed environment (MacMahon et al., 2006; Chen and Mooney, 2011). Fueled by the advances in image representation learning (He et al., 2016), the environments became more realistic by using real-world panorama images of indoor locations (Anderson et al., 2018; Ku et al., 2020). Complementary outdoor environments contain street level panoramas connected by a real-world street layout (Mirowski et al., 2018; Chen et al., 2019; Mehta et al., 2020). Agents in this outdoor environment are trained to follow human written navigation instructions (Chen et al., 2019; Xiang et al., 2020), instructions generated by Google Maps (Hermann et al., 2020), or a combination of both (Zhu et al., 2021b). Recent work focuses on analyzing the navigation agents by introducing better trajectory overlap metrics (Jain et al., 2019; Ilharco et al., 2019) or diagnosing the performance under certain constraints such as uni-modal inputs (Thomason et al., 2019) and masking direction or object tokens (Zhu et al., 2021a). Other work used a trained VLN agent to evaluate automatically generated navigation instructions (Zhao et al., 2021). An open problem in indoor VLN is

	Seen								Unseen							
	Merged				Touchdown		map2seq		Merged				Touchdown		map2seq	
	dev		test		test		test		dev		test		test		test	
Model	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC
best non-merged	-	-	-	-	**44.9**	**29.1**	62.3	46.7	-	-	-	-	21.6	14.9	42.2	30.3
ORAR full model																
• no image	37.5	26.6	35.8	24.7	23.0	14.8	58.3	42.1	31.6	22.3	27.0	19.2	16.6	11.7	**46.5**	**33.2**
• ResNet pre-final	51.3	**38.8**	49.3	**36.8**	39.1	27.7	**67.3**	**52.8**	28.9	22.0	25.7	20.0	17.4	13.6	41.3	32.1
• ResNet 4th-to-last	**53.4**	**37.8**	**51.8**	**35.7**	**46.0**	**30.1**	62.1	45.5	**35.7**	**25.4**	**33.6**	**24.2**	**27.0**	**19.3**	**46.1**	**33.5**

Table 5: Results for models trained on the merged dataset. Test results are presented for the merged test set and individual Touchdown and map2seq test sets. Metrics are normalized Dynamic Time Warping (nDTW) and task completion (TC). In the first row the best results of Table 2 (non-merged training sets) are listed for comparison. The bottom section presents results on the *ORAR full model* with different types of image features.

the generalization of navigation performance to previously unseen areas. Proposed solutions include back translation with environment dropout (Tan et al., 2019), multi-modal environment representation (Hu et al., 2019) or semantic segmented images (Zhang et al., 2020). Notably the latter work identifies the same problem in the Touchdown task.

8 Conclusion

We presented an investigation of outdoor vision and language navigation in seen and unseen environments. We introduced the heading delta feature and junction type embedding to correct an artifact of the environment and explicitly model the number of outgoing edges, respectively. Both are helpful to boost and analyze performance in the unseen scenario. We conducted experiments on two datasets and showed that the considered visual features poorly generalize to unseen areas. We conjecture that VLN tasks need to grow in scale and diversity of geographical environments and navigation tasks.

Acknowledgments

The research reported in this paper was supported by a Google Focused Research Award on "Learning to Negotiate Answers in Multi-Pass Semantic Parsing".

References

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 (CVPR)*, Salt Lake City, Utah.

Devendra Singh Chaplot, Kanthashree Mysore Sathyendra, Rama Kumar Pasumarthi, Dheeraj Rajagopal, and Ruslan Salakhutdinov. 2018. Gated-attention architectures for task-oriented language grounding. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, New Orleans, Louisiana.

David L. Chen and Raymond J. Mooney. 2011. Learning to interpret natural language navigation instructions from observations. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, San Francisco, California.

Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. 2019. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, California.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, Minneapolis, Minnesota.

Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. 2005. Bidirectional lstm networks for improved phoneme classification and recognition. In *Proceedings of International Conference on Artificial Neural Networks: Formal Models and Their Applications (ICANN)*, Warsaw, Poland.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, Nevada.

Karl Moritz Hermann, Mateusz Malinowski, Piotr Mirowski, Andras Banki-Horvath, Keith Anderson, and Raia Hadsell. 2020. Learning to follow directions in street view. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, New York, New York.

Ronghang Hu, Daniel Fried, Anna Rohrbach, Dan Klein, Trevor Darrell, and Kate Saenko. 2019. Are you looking? grounding to multiple modalities in vision-and-language navigation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6551–6557, Florence, Italy. Association for Computational Linguistics.

Gabriel Ilharco, Vihan Jain, Alexander Ku, Eugene Ie, and Jason Baldridge. 2019. Effective and general evaluation for instruction conditioned navigation using dynamic time warping. In *NeurIPS Visually Grounded Interaction and Language Workshop (ViGIL)*, Vancouver, Canada.

Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. 2019. Stay on the path: Instruction fidelity in vision-and-language navigation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, Florence, Italy.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, California.

Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. 2020. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *Computer Vision – ECCV 2020*, Cham.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. 2020. Room-Across-Room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Conference on Empirical Methods for Natural Language Processing (EMNLP)*, Online.

Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. 2006. Walk the talk: Connecting language, knowledge, and action in route instructions. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)*, Boston, Massachusetts.

Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. 2020. Improving vision-and-language navigation with image-text pairs from the web. In *Proceedings of the European Conference on Computer Vision (ECCV)*, Glasgow, UK,.

Harsh Mehta, Yoav Artzi, Jason Baldridge, Eugene Ie, and Piotr Mirowski. 2020. Retouchdown: Releasing touchdown on StreetLearn as a public resource for language grounding tasks in street view. In *Proceedings of the Third International Workshop on Spatial Language Understanding (SpLU)*, Online.

Piotr Mirowski, Matthew Koichi Grimes, Mateusz Malinowski, Karl Moritz Hermann, Keith Anderson, Denis Teplyashin, Karen Simonyan, Koray Kavukcuoglu, Andrew Zisserman, and Raia Hadsell. 2018. Learning to navigate in cities without a map. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NeurIPS)*, Montréal, Canada.

Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, and Peter Kontschieder. 2017. The mapillary vistas dataset for semantic understanding of street scenes. In *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2020. BPE-dropout: Simple and effective subword regularization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, Online.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115:211–252.

Raphael Schumann and Stefan Riezler. 2021. Generating landmark navigation instructions from maps as a graph-to-text problem. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*, Online.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, Berlin, Germany.

Hao Tan, Licheng Yu, and Mohit Bansal. 2019. Learning to navigate unseen environments: Back translation with environmental dropout. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT))*, Minneapolis, Minnesota.

Jesse Thomason, Daniel Gordon, and Yonatan Bisk. 2019. Shifting the baseline: Single modality performance on visual navigation & QA. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, Minneapolis, Minnesota.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NeurIPS)*, Long Beach, California.

Jiannan Xiang, Xin Wang, and William Yang Wang. 2020. Learning to stop: A simple yet effective approach to urban vision-language navigation. In *Findings of the Association for Computational Linguistics (ACL Findings)*, Online.

Kailun Yang, Xinxin Hu, Yicheng Fang, Kaiwei Wang, and Rainer Stiefelhagen. 2020. Omnisupervised omnidirectional semantic segmentation. *IEEE Transactions on Intelligent Transportation Systems (T-ITS)*.

Yubo Zhang, Hao Tan, and Mohit Bansal. 2020. Diagnosing the environment bias in vision-and-language navigation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI)*, Yokohama, Japan.

Ming Zhao, Peter Anderson, Vihan Jain, Su Wang, Alexander Ku, Jason Baldridge, and Eugene Ie. 2021. On the evaluation of vision-and-language navigation instructions. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Online.

Wanrong Zhu, Yuankai Qi, Pradyumna Narayana, Kazoo Sone, Sugato Basu, Xin Eric Wang, Qi Wu, Miguel P. Eckstein, and William Yang Wang. 2021a. Diagnosing vision-and-language navigation: What really matters. *CoRR*, abs/2103.16561.

Wanrong Zhu, Xin Wang, Tsu-Jui Fu, An Yan, Pradyumna Narayana, Kazoo Sone, Sugato Basu, and William Yang Wang. 2021b. Multimodal text style transfer for outdoor vision-and-language navigation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Online.

	Touchdown		map2seq	
Ablation	**dev**	**test**	**dev**	**test**
ORAR full model	**29.9**	**29.1**	**49.1**	**46.7**
- no 2nd RNN	23.2	23.9	43.3	40.6
- no BPE dropout	26.6	25.9	45.2	43.1
- no text attention	9.4	10.4	22.0	21.8
- no image attention	21.5	20.1	**48.8**	**45.7**

Table 6: ORAR full model ablation study on the **seen** scenario of Touchdown and map2seq. Metric is task completion and ablations are not cumulative.

A Architecture Ablation

We perform ablation studies on the ORAR full model in the seen scenario to measure the impact of individual architecture components. As seen in Table 6, removing the second decoder RNN layer or BPE dropout results in a decrease of six and three task completion points, respectively. The largest drop in performance is observed when removing the text attention mechanism. This again shows the importance of attention over the encoder in sequence-to-sequence models. Removing the image attention mechanism on the other hand does not affect task completion on the map2seq dataset.

B Mixed-Model

The findings in Section 6.2 inspire us to modify the ORAR model to use distinct visual features for the orientation and directions/stopping task. The orientation task is equivalent to the very first action prediction by the agent. Thus we modify the model architecture to use the ResNet 4th-to-last features (+text representation) to predict the first action and then start the recurrent prediction of the remaining actions with a different set of visual features (pre-final for the seen scenario and no image features for the unseen scenario). The results for this **ORAR mixed** model trained on the merged dataset are shown in Table 7. We only test it on Touchdown because map2seq does not have the orientation task. The mixed model significantly outperforms the single visual feature model on the Touchdown seen test set but unfortunately shows no improvement in the unseen scenario.

C Additional Metrics and Individual Runs

We present the results of the individual repetitions and additional metrics for the main results in Table 2 and the results on the merged dataset in Table 5. The additional metrics are success weighted normalized Dynamic Time Warping (Ilharco et al., 2019) and shortest-path distance (Chen et al., 2019).

	Seen								Unseen							
	Merged				Touchdown		map2seq		Merged				Touchdown		map2seq	
	dev		test		test		test		dev		test		test		test	
Model	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC	nDTW	TC
best non-merged	-	-	-	-	44.9	29.1	62.3	46.7	-	-	-	-	21.6	14.9	42.2	30.3
best merged	53.4	37.8	51.8	35.7	46.0	30.1	**67.3**	**52.8**	**35.7**	**25.4**	**33.6**	**24.2**	**27.0**	**19.3**	**46.1**	**33.5**
	• 4th-to-last + pre-final								• 4th-to-last + no image							
ORAR mixed model	**58.6**	**44.4**	**57.4**	**42.9**	**51.3**	**36.9**	-	-	**36.3**	**26.1**	**33.6**	**23.9**	**26.3**	**18.3**	-	-

Table 7: Results for the mixed model in comparison to previous best results. Metrics are normalized Dynamic Time Warping (nDTW) and task completion (TC). In the first two rows the best results of Table 2 and Table 5 are listed for comparison. The last section presents results for the *ORAR mixed model* which uses different image features for different sub-tasks.

	Seen								Unseen							
	Touchdown				map2seq				Touchdown				map2seq			
	dev		test		dev		test		dev		test		dev		test	
Model	SDTW	SPD	SDTW	SPD	SDTW	SPD	SDTW	SPD	SDTW	SPD	SDTW	SPD	SDTW	SPD	SDTW	SPD
RConcat	9.8	20.4	11.1	20.4	16.0	19.0	13.7	20.1	1.8	29.6	1.4	29.3	1.2	33.1	1.7	34.1
GA	11.1	18.7	10.9	19.0	17.2	16.5	16.0	18.0	1.3	31.0	1.7	30.5	1.4	34.3	1.3	34.3
ARC	14.1	18.6	13.5	19.4	-	-	-	-	-	-	-	-	-	-	-	-
ARC+l2s	19.0	17.1	16.3	18.8	-	-	-	-	-	-	-	-	-	-	-	-
VLN Transformer	12.9	21.5	14.0	21.2	17.5	18.6	15.9	19.0	1.9	29.5	2.3	29.6	-	-	-	-
ORAR full model																
• ResNet pre-final	24.5	15.0	23.8	16.2	46.7	5.9	44.4	6.6	8.6	26.7	7.6	26.7	22.3	15.6	22.8	16.3
• ResNet 4th-to-last	28.3	11.1	27.4	11.7	41.1	7.2	39.5	7.6	14.3	20.0	13.6	20.7	25.8	11.9	28.3	12.7
ORAR full model	• ResNet 4th-to-last				• ResNet pre-final				• ResNet 4th-to-last				• ResNet 4th-to-last			
- no heading delta	28.3	10.9	27.6	11.5	45.4	6.8	42.7	7.7	14.0	20.5	13.5	20.8	20.4	16.8	21.9	17.1
- no junction type	23.1	13.6	22.8	13.9	47.2	7.6	43.0	8.6	4.0	26.6	3.7	26.7	4.3	28.9	4.2	29.9

Table 8: Results on Touchdown and map2seq for the seen and unseen scenario. Metrics are success weighted normalized Dynamic Time Warping (SDTW) and shortest-path distance (SPD). For SDTW higher values are better and for SPD lower values are better.

	Seen												Unseen											
	task completion of the ten repetitions										mean	std	task completion of the ten repetitions										mean	std
ORAR full model																								
• ResNet pre-final	26.1	18.5	25.8	25.1	26.8	28.7	24.4	25.5	25.6	26.0	25.3	2.5	8.8	9.2	7.3	9.8	8.5	8.4	10.0	8.2	9.4	8.1	8.8	0.8
• ResNet 4th-to-last	28.2	30.0	26.9	29.6	27.4	29.2	30.4	30.0	28.3	30.7	29.1	1.2	12.0	15.1	14.5	15.5	14.3	16.0	16.5	14.9	14.5	15.3	14.9	1.2
ORAR full model	• ResNet 4th-to-last												• ResNet 4th-to-last											
- no heading delta	29.2	30.0	27.4	29.9	29.0	29.5	31.2	29.3	28.4	29.0	29.3	1.0	14.7	13.7	15.5	14.9	14.1	13.5	16.0	15.1	16.0	14.5	14.8	0.8
- no junction type	24.1	24.5	22.6	21.9	24.4	25.7	26.1	24.5	24.5	24.1	24.2	1.2	4.4	5.0	4.2	4.2	4.2	3.8	4.5	4.2	5.1	4.3	4.4	0.4

Table 9: Task completion for the ten individual runs with mean and standard deviation on the Touchdown seen and unseen test set.

	Seen												Unseen											
	task completion of the ten repetitions										mean	std	task completion of the ten repetitions										mean	std
ORAR full model																								
• ResNet pre-final	41.0	48.8	47.8	47.9	45.8	49.5	45.8	48.2	44.6	47.4	46.7	2.4	22.4	18.8	26.0	24.5	26.1	28.1	22.1	26.8	24.4	26.6	24.6	2.6
• ResNet 4th-to-last	40.5	42.2	42.1	42.1	38.6	42.9	41.2	42.1	45.2	40.5	41.7	1.6	32.9	29.6	28.9	28.5	27.6	32.2	26.8	33.6	34.0	28.4	30.3	2.5
ORAR full model	• ResNet pre-final												• ResNet 4th-to-last											
- no heading delta	46.0	43.1	47.1	47.5	45.0	48.4	36.2	44.6	47.1	43.8	44.9	3.3	23.2	24.0	21.6	25.8	24.5	23.6	23.8	23.2	22.0	24.5	23.6	1.2
- no junction type	44.9	46.1	46.2	44.0	43.2	46.5	44.9	47.1	45.5	42.1	45.1	1.5	5.1	4.5	5.0	5.1	4.6	3.9	5.6	3.8	4.4	4.6	4.7	0.5

Table 10: Task completion for the ten individual runs with mean and standard deviation on the map2seq seen and unseen test set.

	Seen								Unseen							
	Merged				Touchdown		map2seq		Merged				Touchdown		map2seq	
	dev		test		test		test		dev		test		test		test	
Model	SDTW	SPD	SDTW	SPD	SDTW	SPD	SDTW	SPD	SDTW	SPD	SDTW	SPD	SDTW	SPD	SDTW	SPD
ORAR full model																
• no image	25.0	18.8	23.2	19.4	13.9	26.1	39.8	7.8	20.6	17.9	17.7	21.3	10.5	26.7	31.1	11.4
• ResNet pre-final	36.8	12.5	34.8	14.1	26.1	18.8	50.2	5.7	20.3	20.1	18.4	22.0	12.2	25.8	30.2	14.8
• ResNet 4th-to-last	35.9	9.3	33.8	9.8	28.4	11.7	43.2	6.5	23.6	14.9	22.5	16.6	17.7	19.2	31.4	11.7
ORAR mixed model																
• 4th-to-last + pre-final	42.1	8.6	40.8	9.3	34.8	11.5	-	-	-	-	-	-	-	-	-	-
• 4th-to-last + no image	-	-	-	-	-	-	-	-	24.1	15.1	22.2	17.2	16.9	20.4	-	-

Table 11: Results for models trained on the merged dataset. Test results are presented for the merged test set and individual Touchdown and map2seq test sets. Metrics are success weighted normalized Dynamic Time Warping (SDTW) and shortest-path distance (SPD). For SDTW higher values are better and for SPD lower values are better.

	Seen												Unseen											
	task completion of the ten repetitions										mean	std	task completion of the ten repetitions										mean	std
ORAR full model																								
• no image	24.0	24.1	25.3	25.8	24.6	26.1	24.4	24.1	24.1	24.5	24.7	0.7	20.1	18.2	19.5	19.7	18.6	18.6	18.7	19.8	18.6	19.9	19.2	0.7
• ResNet pre-final	36.7	34.7	35.3	37.1	36.4	36.1	38.8	36.4	36.8	39.5	36.8	1.4	18.9	19.8	19.8	20.8	20.1	20.0	20.5	19.9	20.2	19.9	20.0	0.5
• ResNet 4th-to-last	34.9	36.2	35.4	35.4	36.2	36.5	34.1	36.3	36.3	35.5	35.7	0.7	23.8	24.9	25.8	23.9	24.1	23.4	24.7	23.9	23.5	24.2	24.2	0.7
ORAR mixed model																								
• 4th-to-last + pre-final	43.8	42.6	43.4	43.2	43.6	42.0	44.1	42.1	41.9	42.7	42.9	0.8	-	-	-	-	-	-	-	-	-	-	-	-
• 4th-to-last + no image	-	-	-	-	-	-	-	-	-	-	-	-	23.8	23.4	23.7	23.4	24.1	24.7	24.6	24.1	23.8	22.9	23.8	0.5

Table 12: Task completion for the ten individual runs with mean and standard deviation on the merged seen and unseen test set.

	Seen												Unseen											
	task completion of the ten repetitions										mean	std	task completion of the ten repetitions										mean	std
ORAR full model																								
• no image	14.1	14.4	15.8	16.5	14.1	16.3	14.5	14.9	13.3	14.5	14.8	1.0	12.1	10.7	12.1	12.2	11.0	11.5	11.5	12.9	11.0	12.2	11.7	0.7
• ResNet pre-final	27.5	25.1	26.4	28.4	26.6	27.4	30.3	27.7	27.0	30.2	27.7	1.5	13.1	13.0	13.1	14.1	13.5	13.7	14.1	13.5	13.9	13.7	13.6	0.4
• ResNet 4th-to-last	30.7	30.4	30.0	30.1	30.0	30.2	29.2	30.2	30.4	30.0	30.1	0.4	18.0	20.3	20.8	18.8	18.9	18.0	20.2	19.8	18.6	19.6	19.3	0.9
ORAR mixed model																								
• 4th-to-last + pre-final	37.6	36.3	36.4	37.8	37.9	35.1	38.0	36.6	35.6	37.4	36.9	1.0	-	-	-	-	-	-	-	-	-	-	-	-
• 4th-to-last + no image	-	-	-	-	-	-	-	-	-	-	-	-	17.8	17.9	18.1	18.8	18.8	19.2	19.2	18.4	17.9	17.1	18.3	0.6

Table 13: Task completion for the ten individual runs with mean and standard deviation on the Touchdown seen and unseen test set, trained on the merged training set.

	Seen												Unseen											
	task completion of the ten repetitions										mean	std	task completion of the ten repetitions										mean	std
ORAR full model																								
• no image	41.5	41.2	42.1	42.1	43.2	43.4	41.9	40.4	43.1	42.1	42.1	0.9	35.1	32.5	33.6	33.8	32.9	32.0	32.4	32.8	32.8	34.2	33.2	0.9
• ResNet pre-final	53.0	51.5	51.0	52.4	53.6	51.5	53.6	51.6	53.9	55.8	52.8	1.4	29.9	32.6	32.2	33.4	32.6	32.0	32.6	32.0	32.0	31.5	32.1	0.9
• ResNet 4th-to-last	42.5	46.4	44.9	44.6	47.1	47.8	42.8	47.1	46.6	45.2	45.5	1.7	34.8	33.5	35.2	33.5	33.9	33.6	33.1	31.6	32.6	33.0	33.5	1.0

Table 14: Task completion for the ten individual runs with mean and standard deviation on the map2seq seen and unseen test set, trained on the merged training set.