Translating Navigation Instructions in Natural Language to a High-Level Plan for Behavioral Robot Navigation

Xiaoxue Zang[1]*, Ashwini Pokle[1]*, Marynel Vázquez[2], Kevin Chen[1], Juan Carlos Niebles[1], Alvaro Soto[3], Silvio Savarese[1]
[1] Stanford University, [2] Yale University, [3] P. Universidad Católica de Chile
[1]{xzang, ashwinipokle, kchen92, jniebles, ssilvio}@stanford.edu,
[2]marynel.vazquez@yale.edu, [3]asoto@ing.puc.cl

Abstract

We propose an end-to-end deep learning model for translating free-form natural language instructions to a high-level plan for behavioral robot navigation. The proposed model uses attention mechanisms to connect information from user instructions with a topological representation of the environment. To evaluate this model, we collected a new dataset for the translation problem containing 11,051 pairs of user instructions and navigation plans. Our results show that the proposed model outperforms baseline approaches on the new dataset. Overall, our work suggests that a topological map of the environment can serve as a relevant knowledge base for translating natural language instructions into a sequence of navigation behaviors.

1 Introduction

Enabling robots to follow navigation instructions in natural language can facilitate human-robot interaction across a variety of applications. For instance, within the service robotics domain, robots can follow navigation instructions to help with mobile manipulation (Tellex et al., 2011) and delivery tasks (Veloso et al., 2015).

Interpreting navigation instructions in natural language is difficult due to the high variability in the way people describe routes (Chen and Mooney, 2011). For example, there are a variety of ways to describe the route in Fig. 1(a):

- *"Exit the room, turn right, follow the corridor until you pass a vase on your left, and enter the next room on your left"*; or
- *"Turn right after you exit the room, and enter the room on the left right before the end of the corridor"*; or
- *"Advance forward to the right after going out of the door. Enter the room which is in the middle of two vases on your left."*

*Both authors contributed equally to this work.

Each fragment of a sentence within these instructions can be mapped to one or more than one navigation behaviors. For instance, assume that a robot counts with a number of primitive, navigation behaviors, such as *"enter the room on the left (or on right)"* , *"follow the corridor"*, *"cross the intersection"*, etc. Then, the fragment *"advance forward"* in a navigation instruction could be interpreted as a *"follow the corridor"* behavior, or as a sequence of *"follow the corridor"* interspersed with *"cross the intersection"* behaviors depending on the topology of the environment. Resolving such ambiguities often requires reasoning about "common-sense" concepts, as well as interpreting spatial information and landmarks, e.g., in sentences such as *"the room on the left right before the end of the corridor"* and *"the room which is in the middle of two vases"*.

In this work, we pose the problem of interpreting navigation instructions as finding a mapping (or grounding) of the commands into an executable navigation plan. While the plan is typically modeled as a formal specification of low-level motions (Chen and Mooney, 2011) or a grammar (Artzi and Zettlemoyer, 2013; Matuszek et al., 2010), we focus specifically on translating instructions to a high-level navigation plan based on a topological representation of the environment. This representation is a *behavioral navigation graph*, as recently proposed by (Sepúlveda et al., 2018), designed to take advantage of the semantic structure typical of human environments. The nodes of the graph correspond to semantically meaningful locations for the navigation task, such as kitchens or entrances to rooms in corridors. The edges are parameterized, visuo-motor behaviors that allow a robot to navigate between neighboring nodes, as illustrated in Fig. 1(b). Under this framework, complex navigation routes can be achieved by sequencing behaviors without an explicit metric representation of the world.

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 2657–2666
Brussels, Belgium, October 31 - November 4, 2018.

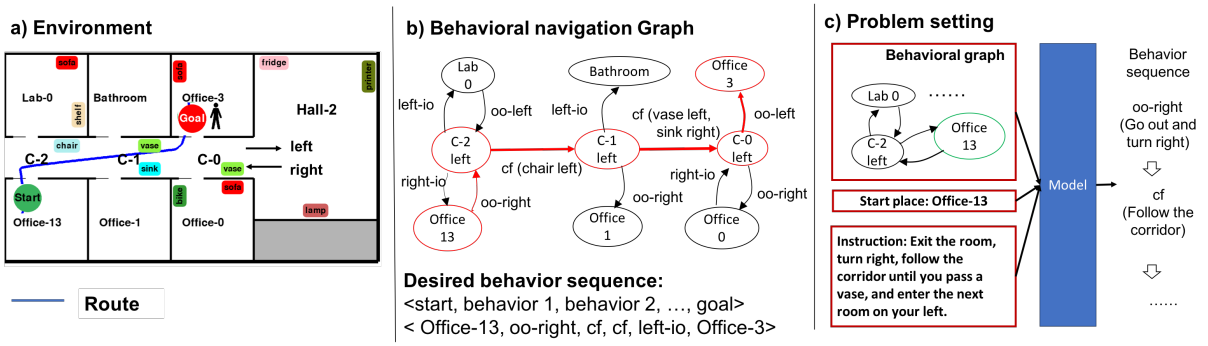

Figure 1: Map of an environment (a), its (partial) behavioral navigation graph (b), and the problem setting of interest (c). The red part of (b) corresponds to the representation of the route highlighted in blue in (a). The codes "oo-left", "oo-right", "cf", "left-io", and "right-io" correspond to the behaviors "go out and turn left", "go out and turn right", "follow the corridor", "enter the room on left", and "enter office on right", respectively.

We formulate the problem of following instructions under the framework of (Sepúlveda et al., 2018) as finding a path in the behavioral navigation graph that follows the desired route, given a known starting location. The edges (behaviors) along this path serve to reach the – sometimes implicit – destination requested by the user. As in (Zang et al., 2018), our focus is on the problem of interpreting navigation directions. We assume that a robot can realize valid navigation plans according to the graph.

We contribute a new end-to-end model for following directions in natural language under the behavioral navigation framework. Inspired by the information retrieval and question answering literature (Lewis and Jones, 1996; Seo et al., 2017; Xiong et al., 2016; Palangi et al., 2016), we propose to leverage the behavioral graph as a knowledge base to facilitate the interpretation of navigation commands. More specifically, the proposed model takes as input user directions in text form, the behavioral graph of the environment encoded as ⟨`node; edge; node`⟩ triplets, and the initial location of the robot in the graph. The model then predicts a set of behaviors to reach the desired destination according to the instructions and the map (Fig. 1(c)). Our main insight is that using attention mechanisms to correlate navigation instructions with the topological map of the environment can facilitate predicting correct navigation plans.

This work also contributes a new dataset of 11,050 pairs of free-form natural language instructions and high-level navigation plans. This dataset was collected through Mechanical Turk using 100 simulated environments with a corresponding topological map and, to the best of our knowledge, it is the first of its kind for behavioral navigation. The dataset opens up opportunities to explore data-driven methods for grounding navigation commands into high-level motion plans.

We conduct extensive experiments to study the generalization capabilities of the proposed model for following natural language instructions. We investigate both generalization to new instructions in known and in new environments. We conclude this paper by discussing the benefits of the proposed approach as well as opportunities for future research based on our findings.

2 Related work

This section reviews relevant prior work on following navigation instructions. Readers interested in an in-depth review of methods to interpret spatial natural language for robotics are encouraged to refer to (Landsiedel et al., 2017).

Typical approaches to follow navigation commands deal with the complexity of natural language by manually parsing commands, constraining language descriptions, or using statistical machine translation methods. While manually parsing commands is often impractical, the first type of approaches are foundational: they showed that it is possible to leverage the compositionality of semantic units to interpret spatial language (Bugmann et al., 2004; Levit and Roy, 2007).

Constraining language descriptions can reduce the size of the input space to facilitate the interpretation of user commands. For example, (Talbot et al., 2016) explored using structured, symbolic language phrases for navigation. As in this earlier work, we are also interested in navigation with a topological map of the environment. However, we do not process symbolic phrases. Our aim is to translate free-form natural language instruc-

tions to a navigation plan using information from a high-level representation of the environment. This translation problem requires dealing with missing actions in navigation instructions and actions with preconditions, such as *"at the end of the corridor, turn right"* (MacMahon et al., 2006).

Statistical machine translation (Koehn, 2009) is at the core of recent approaches to enable robots to follow navigation instructions. These methods aim to automatically discover translation rules from a corpus of data, and often leverage the fact that navigation directions are composed of sequential commands. For instance, (Wong and Mooney, 2006; Matuszek et al., 2010; Chen and Mooney, 2011) used statistical machine translation to map instructions to a formal language defined by a grammar. Likewise, (Kollar et al., 2010; Tellex et al., 2011) mapped commands to spatial description clauses based on the hierarchical structure of language in the navigation problem. Our approach to machine translation builds on insights from these prior efforts. In particular, we focus on end-to-end learning for statistical machine translation due to the recent success of Neural Networks in Natural Language Processing (Goodfellow et al., 2016).

Our work is inspired by methods that reduce the task of interpreting user commands to a sequential prediction problem (Shimizu and Haas, 2009; Mei et al., 2016; Anderson et al., 2018). Similar to Mei et al. and Anderson et al., we use a *sequence-to-sequence* model to enable a mobile agent to follow routes. But instead leveraging visual information to output low-level navigation commands, we focus on using a topological map of the environment to output a high-level navigation plan. This plan is a sequence of behaviors that can be executed by a robot to reach a desired destination (Sepúlveda et al., 2018; Zang et al., 2018).

We explore machine translation from the perspective of automatic question answering. Following (Seo et al., 2017; Xiong et al., 2016), our approach uses attention mechanisms to learn alignments between different input modalities. In our case, the inputs to our model are navigation instructions, a topological environment map, and the start location of the robot (Fig. 1(c)). Our results show that the map can serve as an effective source of contextual information for the translation task. Additionally, it is possible to leverage this kind of information in an end-to-end fashion.

3 Problem Formulation

Our goal is to translate navigation instructions in text form into a sequence of behaviors that a robot can execute to reach a desired destination from a known start location. We frame this problem under a behavioral approach to indoor autonomous navigation (Sepúlveda et al., 2018) and assume that prior knowledge about the environment is available for the translation task. This prior knowledge is a topological map, in the form of a behavioral navigation graph (Fig. 1(b)). The nodes of the graph correspond to semantically-meaningful locations for the navigation task, and its directed edges are visuo-motor behaviors that a robot can use to move between nodes. This formulation takes advantage of the rich semantic structure behind man-made environments, resulting in a compact route representation for robot navigation.

Fig. 1(c) provides a schematic view of the problem setting. The inputs are: (1) a navigation graph m, (2) the starting node s of the robot in m, and (3) a set of free-form navigation instructions I in natural language. The instructions describe a path in the graph to reach from s to a – potentially implicit – destination node g. Using this information, the objective is to predict a suitable sequence of robot behaviors $b_1, \ldots, b_T$ to navigate from s to g according to I. From a supervised learning perspective, the goal is then to estimate:

$$\underset{b_1,\ldots,b_T}{\operatorname{argmax}} P(b_1, \ldots, b_T | m, s, I) \qquad (1)$$

based on a dataset of input-target pairs $\{(\mathbf{x}_i, \mathbf{y}_i) \,|\, 0 \leq i \leq N\}$, where $\mathbf{x}_i = (m, s, I)_i$ and $\mathbf{y}_i = (b_1, \ldots, b_T)_i$, respectively. The sequential execution of the behaviors $b_1, \ldots, b_T$ should replicate the route intended by the instructions I.

We assume no prior linguistic knowledge. Thus, translation approaches have to cope with the semantics and syntax of the language by discovering corresponding patterns in the data.

3.1 The Behavioral Graph: A Knowledge Base For Navigation

We view the behavioral graph m as a knowledge base that encodes a set of navigational rules as triplets $\langle p_i;\, b_l[attr];\, p_j \rangle$, where p_i and p_j are adjacent nodes in the graph, and the edge b_l is an executable behavior to navigate from p_i to p_j. In general, each behaviors includes a list of relevant navigational attributes $attr$ that the robot might encounter when moving between nodes.

Behavior	Description
oo<d>	Go out of the current place and turn <d>
io<d>	Turn <d> and enter the place straight ahead
oio	Exit current place and enter straight ahead
<d>t	Turn <d> at the intersection
cf	Follow (or go straight down) the corridor
sp	Go straight at a T intersection
ch<d>	Cross the hall and turn <d>

Table 1: Behaviors (edges) of the navigation graphs considered in this work. The direction <d> can be left or right.

We consider 7 types of semantic locations, 11 types of behaviors, and 20 different types of landmarks. A location in the navigation graph can be a room, a lab, an office, a kitchen, a hall, a corridor, or a bathroom. These places are labeled with unique tags, such as "room-1" or "lab-2", except for bathrooms and kitchens which people do not typically refer to by unique names when describing navigation routes.

Table 1 lists the navigation behaviors that we consider in this work. These behaviors can be described in reference to visual landmarks or objects, such as paintings, book shelfs, tables, etc. As in Fig. 1, maps might contain multiple landmarks of the same type. Please see the supplementary material (Appendix A) for more details.

4 Approach

We leverage recent advances in deep learning to translate natural language instructions to a sequence of navigation behaviors in an end-to-end fashion. Our proposed model builds on the sequence-to-sequence translation model of (Bahdanau et al., 2015), which computes a soft-alignment between a source sequence (natural language instructions in our case) and the corresponding target sequence (navigation behaviors).

As one of our main contributions, we augment the neural machine translation approach of Bahdanau et al. to take as input not only natural language instructions, but also the corresponding behavioral navigation graph m of the environment where navigation should take place. Specifically, at each step, the graph m operates as a knowledge base that the model can access to obtain information about path connectivity, facilitating the grounding of navigation commands.

Figure 2 shows the structure of the proposed model for interpreting navigation instructions. The model consists of six layers:

Embed layer: The model first encodes each word and symbol in the input sequences I and m into fixed-length representations. The instructions I are embedded into a 100-dimensional pre-trained GloVe vector (Pennington et al., 2014). Each of the triplet components, p_i, $b_l[attr]$, and p_j of the graph m, are one-hot encoded into vectors of dimensionality $2N + E$, where N and E are the number of nodes and edges in m, respectively.

Encoder layer: The model then uses two bidirectional Gated Recurrent Units (GRUs) (Cho et al., 2014) to independently process the information from I and m, and incorporate contextual cues from the surrounding embeddings in each sequence. The outputs of the encoder layer are the matrix $\bar{I} \in \mathbb{R}^{T \times 2H}$ for the navigational commands and the matrix $\bar{G} \in \mathbb{R}^{L \times 2H}$ for the behavioral graph, where H is the hidden size of each GRU, T is the number of words in the instruction I, and L is the number of triplets in the graph m.

Attention layer: Matrices $\bar{I}$ and $\bar{G}$ generated by the encoder layer are combined using an attention mechanism. We use one-way attention because the graph contains information about the whole environment, while the instruction has (potentially incomplete) local information about the route of interest. The use of attention provides our model with a two-step strategy to interpret commands. This resembles the way people find paths on a map: first, relevant parts on the map are selected according to their affinity to each of the words in the input instruction (attention layer); second, the selected parts are connected to assemble a valid path (decoder layer). More formally, let $\bar{G}_i$ ($i \in [1, L]$) be the i-th row of $\bar{G}$, and $\bar{I}_j$ ($j \in [1, T]$) the j-th row of $\bar{I}$. We use each encoded triplet $\bar{G}_i$ in $\bar{G}$ to calculate its associated attention distribution $a_i \in \mathbb{R}^T$ over all the atomic instructions $\bar{I}_j$:

$$e_i = [\bar{G}_i W \bar{I}_1^\intercal, \ldots, \bar{G}_i W \bar{I}_T^\intercal] \quad (2)$$

$$a_i = softmax(e_i) \quad (3)$$

where the matrix $W \in \mathbb{R}^{2H \times 2H}$ serves to combine the different sources of information $\bar{G}$ and $\bar{I}$. Each component a_{ij} of the attention distributions a_i quantifies the affinity between the i-th triplet in $\bar{G}$ and the j-th word in the corresponding input I.

The model then uses each attention distribution a_i to obtain a weighted sum of the encodings of the words in $\bar{I}$, according to their relevance to the corresponding triplet $\bar{G}_i$. This results in L attention vectors $R_i \in \mathbb{R}^{2H}$, $R_i = \sum_{j=1}^{T} a_{ij} I_j$.

The final step in the attention layer concatenates each R_i with $\bar{G}_i$ to generate the outputs

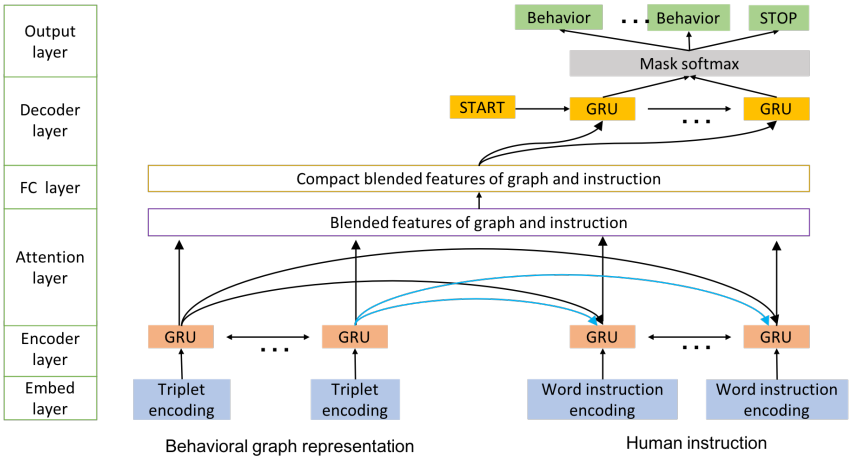

Figure 2: Model overview. The model contains six layers, takes the input of behavioral graph representation, free-form instruction, and the start location (yellow block marked as START in the decoder layer) and outputs a sequence of behaviors.

$F_i = [R_i; \bar{G}_i]$, $i \in [1, L]$. Following (Seo et al., 2017), we include the encoded triplet $\bar{G}_i$ in the output tensor F_i of this layer to prevent early summaries of relevant map information.

FC layer: The model reduces the dimensionality of each individual vector F_i from $4H$ to H with a fully-connected (FC) layer. The resulting L vectors are output to the next layer as columns of a context matrix $C \in \mathbb{R}^{H \times L}$.

Decoder layer: After the FC layer, the model predicts likelihoods over the sequence of behaviors that correspond to the input instructions with a GRU network. Without loss of generality, consider the t-th recurrent cell in the GRU network. This cell takes two inputs: a hidden state vector h_{t-1} from the prior cell, and a one-hot embedding of the previous behavior b_{t-1} that was predicted by the model. Based on these inputs, the GRU cell outputs a new hidden state h_t to compute likelihoods for the next behavior. These likelihoods are estimated by combining the output state h_t with relevant information from the context C:

$$\hat{d}_{ts} = v_a^\intercal \tanh(W_1 h_t + W_2 C_s) \quad (4)$$

$$d_t = softmax(\hat{d}_{t1}, \ldots, \hat{d}_{tL}) \quad (5)$$

where W_1, W_2, and v_a are trainable parameters. The attention vector $d_t \in \mathbb{R}^L$ in Eq. (5) quantifies the affinity of h_t with respect to each of the columns C_s of C, where $s \in [1, L]$. The attention vector also helps to estimate a dynamic contextual vector $S_t = \sum_{s=1}^{L} d_{ts} C_s$ that the t-th GRU cell uses to compute logits for the next behavior:

$$o_t = W_3[S_t; h_t] \quad (6)$$

with W_3 trainable parameters. Note that o_t includes a value for each of the pre-defined behaviors in the graph m, as well as for a special "*stop*" symbol to identify the end of the output sequence.

Output layer: The final layer of the model searches for a valid sequence of robot behaviors based on the robot's initial node, the connectivity of the graph m, and the output logits from the previous decoder layer. Again, without loss of generality, consider the t-th behavior b_t that is finally predicted by the model. The search for this behavior is implemented as:

$$b_t = argmax(softmax(o_t + mask(m, n_t))) \quad (7)$$

with $mask(m, n_t)$ a masking function that takes as input the graph m and the node n_t that the robot reaches after following the sequence of behaviors $b_1, \ldots, b_{t-1}$ previously predicted by the model. The $mask$ function returns a vector of the same dimensionality as the logits o_t, but with zeros for the valid behaviors after the last location n_t and for the special stop symbol, and $-\inf$ for any invalid predictions according to the connectivity of the behavioral navigation graph.

5 Dataset

We created a new dataset for the problem of following navigation instructions under the behavioral navigation framework of (Sepúlveda et al., 2018).[1] This dataset was created using Amazon Mechanical Turk and 100 maps of simulated indoor environments, each with 6 to 65 rooms. To the best of our knowledge, this is the first bench-

[1]The dataset is publicly available through the website: `follow-nav-directions.stanford.edu`.

Dataset	# Single	# Double	Total
Training	4062	2002	8066
Test-Repeated	944	34	1012
Test-New	962	0	962

Table 2: Dataset statistics. "# Single" indicates the number of navigation plans with a single natural language instruction. "# Double" is the number of plans with two different instructions. The total number of plans is (# Single) $\times$ 2(# Double).

mark for comparing translation models in the context of behavioral robot navigation.

As shown in Table 2, the dataset consists of 8066 pairs of free-form natural language instructions and navigation plans for training. This training data was collected from 88 unique simulated environments, totaling 6064 distinct navigation plans (2002 plans have two different navigation instructions each; the rest has one). The dataset contains two test set variants:

1) Test-Repeated: Contains 1012 pairs of instructions and navigation plans. These routes are not part of the training set; however, they are collected using environments that are part of the training set.

2) Test-New: Contains 962 pairs of instructions and navigation plans. This test set is more challenging than the Test-Repeated dataset because it contains new routes on 12 new indoor environments not included in the training set.

While the dataset was collected with simulated environments, no structure was imposed on the navigation instructions while crowd-sourcing data. Thus, many instructions in our dataset are ambiguous. Moreover, the order of the behaviors in the instructions is not always the same. For instance, a person said *"turn right and advance"* to describe part of a route, while another person said *"go straight after turning right"* in a similar situation. The high variability present in the natural language descriptions of our dataset makes the problem of decoding instructions into behaviors not trivial. See Appendix A of the supplementary material for additional details on our data collection effort.

6 Experiments

This section describes our evaluation of the proposed approach for interpreting navigation commands in natural language. We provide both quantitative and qualitative results.

6.1 Evaluation Metrics

While computing evaluation metrics, we only consider the behaviors present in the route because they are sufficient to recover the high-level navigation plan from the graph. Our metrics treat each behavior as a single token. For example, the sample plan "R-1 oor C-1 cf C-1 lt C-0 cf C-0 iol O-3" is considered to have 5 tokens, each corresponding to one of its behaviors ("oor", "cf", "lt", "cf", "iol"). In this plan, "R-1","C-1", "C-0", and "O-3" are symbols for locations (nodes) in the graph.

We compare the performance of translation approaches based on four metrics:

- Exact Match (EM). As in (Shimizu and Haas, 2009), EM is 1 if a predicted plan matches exactly the ground truth; otherwise it is 0.

- F1 score (F1). The harmonic average of the precision and recall over all the test set (Chinchor and Sundheim, 1993).

- Edit Distance (ED). The minimum number of insertions, deletions or swap operations required to transform a predicted sequence of behaviors into the ground truth sequence (Navarro, 2001).

- Goal Match (GM). GM is 1 if a predicted plan reaches the ground truth destination (even if the full sequence of behaviors does not match exactly the ground truth). Otherwise, GM is 0.

6.2 Models Used in the Evaluation

We compare the proposed approach for translating natural language instructions into a navigation plan against alternative deep-learning models:

Baseline model. The baseline approach is based on (Shimizu and Haas, 2009). It divides the task of interpreting commands for behavioral navigation into two steps: path generation, and path verification. For path generation, this baseline uses a standard sequence-to-sequence model augmented with an attention mechanism, similar to (Bahdanau et al., 2015; Zang et al., 2018). For path verification, the baseline uses depth-first search to find a route in the graph that matches the sequence of predicted behaviors. If no route matches perfectly, the baseline changes up to three behaviors in the predicted sequence to try to turn it into a valid path.

Ablation model. To test the impact of using the behavioral graphs as an extra input to our translation model, we implemented a version of our

approach that only takes natural language instructions as input. In this ablation model, the output of the bidirectional GRU that encodes the input instruction I is directly fed to the decoder layer. This model does not have the attention and FC layers described in Sec. 4, nor uses the masking function in the output layer.

Ablation with mask model. This model is the same as the previous Ablation model, but with the masking function in the output layer.

6.3 Implementation Details

We pre-processed the inputs to the various models that are considered in our experiment. In particular, we lowercased, tokenized, spell-checked and lemmatized the input instructions in text-form using WordNet (Miller, 1995). We also truncated the graphs to a maximum of 300 triplets, and the navigational instructions to a maximum of 150 words. Only 6.4% (5.4%) of the unique graphs in the training (validation) set had more than 300 triplets, and less than 0.15% of the natural language instructions in these sets had more than 150 tokens.

The dimensionality of the hidden state of the GRU networks was set to 128 in all the experiments. In general, we used 12.5% of the training set as validation for choosing models' hyperparameters. In particular, we used dropout after the encoder and the fully-connected layers of the proposed model to reduce overfitting. Best performance was achieved with a dropout rate of 0.5 and batch size equal to 256. We also used scheduled sampling (Bengio et al., 2015) at training time for all models except the baseline.

We input the triplets from the graph to our proposed model in alphabetical order, and consider a modification where the triplets that surround the start location of the robot are provided first in the input graph sequence. We hypothesized that such rearrangement would help identify the starting location (node) of the robot in the graph. In turn, this could facilitate the prediction of correct output sequences. In the remaining of the paper, we refer to models that were provided a rearranged graph, beginning with the starting location of the robot, as models with "Ordered Triplets".

6.4 Quantitative Evaluation

Table 3 shows the performance of the models considered in our evaluation on both test sets. The next two sections discuss the results in detail.

6.4.1 Performance in the Test-Repeated Set

First, we can observe that the final model "Ours with Mask and Ordered Triplets" outperforms the Baseline and Ablation models on all metrics in previously seen environments. The difference in performance is particularly evident for the Exact Match and Goal Match metrics, with our model increasing accuracy by 35% and 25% in comparison to the Baseline and Ablation models, respectively. These results suggest that providing the behavioral navigation graph to the model and allowing it to process this information as a knowledge base in an end-to-end fashion is beneficial.

We can also observe from Table 3 that the masking function of Eq. (7) tends to increase performance in the Test-Repeated Set by constraining the output sequence to a valid set of navigation behaviors. For the Ablation model, using the masking function leads to about 10% increase in EM and GM accuracy. For the proposed model (with or without reordering the graph triplets), the increase in accuracy is around 4%. Note that the impact of the masking function is less evident in terms of the F1 score because this metric considers if a predicted behavior exists in the ground truth navigation plan, irrespective of its specific position in the output sequence.

The results in the last four rows of Table 3 suggest that ordering the graph triplets can facilitate predicting correct navigation plans in previously seen environments. Providing the triplets that surround the starting location of the robot first to the model leads to a boost of 4% in EM and GM performance. The rearrangement of the graph triplets also helps to reduce ED and increase F1.

Lastly, it is worth noting that our proposed model (last row of Table 3) outperforms all other models in previously seen environments. In particular, we obtain over 4% increase in EM and GM between our model and the next best two models.

6.4.2 Performance in the Test-New Set

The previous section evaluated model performance on new instructions (and corresponding navigation plans) for environments that were previously seen at training time. Here, we examine whether the trained models succeed on environments that are completely new.

The evaluation on the Test-New Set helps understand the generalization capabilities of the models under consideration. This experiment is more challenging than the one in the previous sec-

Model	Test-Repeated Set				Test-New Set			
	EM ↑	F1 ↑	ED ↓	GM ↑	EM ↑	F1↑	ED ↓	GM ↑
Baseline	25.30	79.83	2.53	26.28	25.44	81.38	2.39	25.44
Ablation	36.36	90.28	1.36	36.36	24.82	88.65	1.71	24.92
Ablation with Mask	45.95	90.08	1.20	46.05	36.45	88.31	1.45	36.56
Ours without Mask	52.47	91.74	0.95	53.95	21.94	87.50	1.78	22.65
Ours with Mask	57.31	91.91	0.91	57.31	38.52	88.98	1.32	38.52
Ours without Mask and with Ordered Triplets	57.21	93.37	0.79	57.71	33.36	**91.02**	1.37	33.78
Ours with Mask and Ordered Triplets	**61.17**	**93.54**	**0.75**	**61.36**	**41.71**	90.22	**1.22**	**41.81**

Table 3: Performance of different models on the test datasets. EM and GM report percentages, and ED corresponds to average edit distance. The symbol ↑ indicates that higher results are better in the corresponding column; ↓ indicates that lower is better.

tion, as can be seen in performance drops in Table 3 for the new environments. Nonetheless, the insights from the previous section still hold: masking in the output layer and reordering the graph triplets tend to increase performance.

Even though the results in Table 3 suggest that there is room for future work on decoding natural language instructions, our model still outperforms the baselines by a clear margin in new environments. For instance, the difference between our model and the second best model in the Test-New set is about 3% EM and GM. Note that the average number of actions in the ground truth output sequences is 7.07 for the Test-New set. Our model's predictions are just 1.22 edits off on average from the correct navigation plans.

6.5 Qualitative Evaluation

This section discusses qualitative results to better understand how the proposed model uses the navigation graph.

6.5.1 Attention Visualization

We analyze the evolution of the attention weights d_t in Eq. (5) to assess if the decoder layer of the proposed model is attending to the correct parts of the behavioral graph when making predictions. Fig 3(b) shows an example of the resulting attention map for the case of a correct prediction. In the Figure, the attention map is depicted as a scaled and normalized 2D array of color codes. Each column in the array shows the attention distribution d_t used to generate the predicted output at step t. Consequently, each row in the array represents a triplet in the corresponding behavioral graph. This graph consists of 72 triplets for Fig 3(b).

We observe a locality effect associated to the attention coefficients corresponding to high values (bright areas) in each column of Fig 3(b). This suggests that the decoder is paying attention to graph triplets associated to particular neighborhoods of the environment in each prediction step. We include additional attention visualizations in the supplementary Appendix, including cases where the dynamics of the attention distribution are harder to interpret.

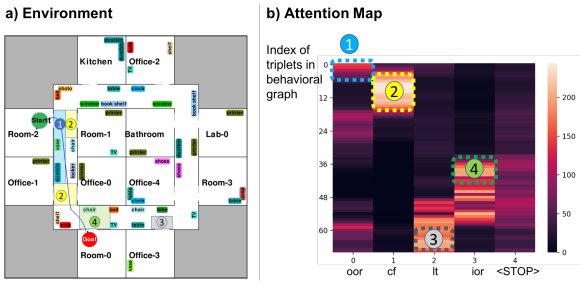

Figure 3: Visualization of the attention weights of the decoder layer. The color-coded and numbered regions on the map (left) correspond to the triplets that are highlighted with the corresponding color in the attention map (right).

6.5.2 Experiments with Sub-Optimal Paths

All the routes in our dataset are the shortest paths from a start location to a given destination. Thus, we collected a few additional natural language instructions to check if our model was able to follow navigation instructions describing sub-optimal paths. One such example is shown in Fig. 4, where the blue route (shortest path) and the red route (alternative path) are described by:

– **Blue route:** "*Go out the office and make a left. Turn right at the corner and go down the hall. Make a right at the next corner and enter the kitchen in front of table.*"

– **Red route:** "*Exit the room 0 and turn right, go to the end of the corridor and turn left, go straight to the end of the corridor and turn left again. After passing bookshelf on your left and table on your right, Enter the kitchen on your right.*"

For both routes, the proposed model was able to predict the correct sequence of navigation behaviors. This result suggests that the model is indeed using the input instructions and is not just approximating shortest paths in the behavioral graph.

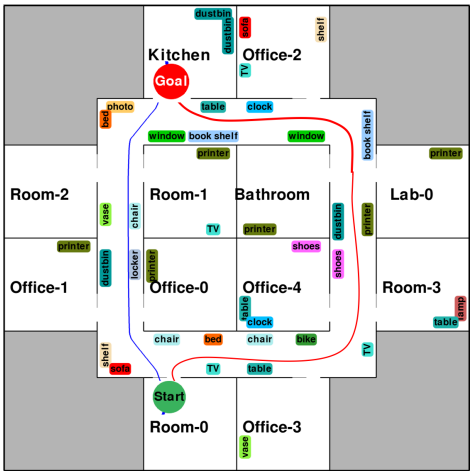

Figure 4: An example of two different navigation paths between the same pair of start and goal locations.

Other examples on the prediction of sub-obtimal paths are described in the Appendix.

7 Conclusion

This work introduced behavioral navigation through free-form natural language instructions as a challenging and a novel task that falls at the intersection of natural language processing and robotics. This problem has a range of interesting cross-domain applications, including information retrieval.

We proposed an end-to-end system to translate user instructions to a high-level navigation plan. Our model utilized an attention mechanism to merge relevant information from the navigation instructions with a behavioral graph of the environment. The model then used a decoder to predict a sequence of navigation behaviors that matched the input commands.

As part of this effort, we contributed a new dataset of 11,051 pairs of user instructions and navigation plans from 100 different environments. Our model achieved the best performance in this dataset in comparison to a two-step baseline approach for interpreting navigation instructions, and a sequence-to-sequence model that does not consider the behavioral graph. Our quantitative and qualitative results suggest that attention mechanisms can help leverage the behavioral graph as a relevant knowledge base to facilitate the translation of free-form navigation instructions. Overall, our approach demonstrated practical form of learning for a complex and useful task.

In future work, we are interested in investigating mechanisms to improve generalization to new environments. For example, pointer and graph networks (Vinyals et al., 2015; Defferrard et al., 2016) are a promising direction to help supervise translation models and predict motion behaviors.

Acknowledgments

The Toyota Research Institute (TRI) provided funds to assist with this research, but this paper solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity. This work is also partially funded by Fondecyt grant 1181739, Conicyt, Chile. The authors would also like to thank Gabriel Sepúlveda for his assistance with parts of this project.

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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yoav Artzi and Luke Zettlemoyer. 2013. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association of Computational Linguistics (TACL)*, 1:49–62.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations (ICLR)*.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179.

Guido Bugmann, Ewan Klein, Stanislao Lauria, and Theocharis Kyriacou. 2004. Corpus-based robotics: A route instruction example. In *Proceedings of Intelligent Autonomous Systems (IAS)*, pages 96–103. Citeseer.

David L. Chen and Raymond J. Mooney. 2011. Learning to interpret natural language navigation instructions from observations. In *AAAI Conference on Artificial Intelligence*, pages 859–865.

Nancy Chinchor and Beth Sundheim. 1993. Muc-5 evaluation metrics. In *Proceedings of the 5th conference on Message understanding*, pages 69–78. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merrienboer, aglar Gülehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder

for statistical machine translation. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *CoRR*, abs/1606.09375.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

P. Koehn. 2009. *Statistical Machine Translation*. Cambridge University Press.

T. Kollar, S. Tellex, D. Roy, and N. Roy. 2010. Toward understanding natural language directions. In *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 259–266.

Christian Landsiedel, Verena Rieser, Matthew Walter, and Dirk Wollherr. 2017. A review of spatial reasoning and interaction for real-world robotics. *Advanced Robotics*, 31(5):222–242.

Michael Levit and Deb Roy. 2007. Interpretation of spatial language in a map navigation task. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 37(3):667–679.

David D Lewis and Karen Spärck Jones. 1996. Natural language processing for information retrieval. *Communications of the ACM*, 39(1):92–101.

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

C. Matuszek, D. Fox, and K. Koscher. 2010. Following directions using statistical machine translation. In *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 251–258.

Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. 2016. Listen, attend, and walk: Neural mapping of navigational instructions to action sequences. In *National Conference on Artificial Intelligence (AAAI)*, pages 2772–2778.

George A. Miller. 1995. Wordnet: A lexical database for english. *Communications of the ACM*.

Gonzalo Navarro. 2001. A guided tour to approximate string matching. *ACM computing surveys (CSUR)*, 33(1):31–88.

Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *International Conference on Learning Representations ICLR*.

G. Sepúlveda, JC. Niebles, and A. Soto. 2018. A deep learning based behavioral approach to indoor autonomous navigation. In *International Conference on Learning Representations (ICRA)*.

Nobuyuki Shimizu and Andrew R. Haas. 2009. Learning to follow navigational route instructions. In *International Joint Conferences on Artificial Intelligence (IJCAI)*.

Ben Talbot, Obadiah Lam, Ruth Schulz, Feras Dayoub, Ben Upcroft, and Gordon Wyeth. 2016. Find my office: Navigating real space from semantic descriptions. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5782–5787.

Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth J Teller, and Nicholas Roy. 2011. Understanding natural language commands for robotic navigation and mobile manipulation. In *National Conference on Artificial Intelligence (AAAI)*, volume 1, page 2.

Manuela M Veloso, Joydeep Biswas, Brian Coltin, and Stephanie Rosenthal. 2015. Cobots: Robust symbiotic autonomous mobile service robots. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, page 4423.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2692–2700.

Yuk Wah Wong and Raymond J Mooney. 2006. Learning for semantic parsing with statistical machine translation. In *Proc. of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 439–446.

Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic memory networks for visual and textual question answering. In *International Conference on Machine Learning (ICML)*, pages 2397–2406.

Xiaoxue Zang, Marynel Vázquez, Juan Carlos Niebles, Alvaro Soto, and Silvio Savarese. 2018. Behavioral indoor navigation with natural language directions. In *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 283–284.