Communication with Robots using Multilayer Recurrent Networks

Bedřich Pišl and **David Mareček**
Charles University,
Faculty of Mathematics and Physics,
Institute of Formal and Applied Linguistics
bedapisl@gmail.com, marecek@ufal.mff.cuni.cz

Abstract

In this paper, we describe an improvement on the task of giving instructions to robots in a simulated block world using unrestricted natural language commands.

1 Introduction

Many of the recent methods for interpreting natural language commands are based mainly on semantic parsers and hand designed rules. This is often due to small datasets, such as Robot Commands Treebank (Dukes, 2013) or datasets by MacMahon et al. (2006) or Han and Schlangen (2017).

Tellex et al. (2011) and Walter et al. (2015) present usage of such systems in real world. They developed a robotic forklift which is able to understand simple natural language commands. For training, they created small dataset by manually annotating the data from Amazon Mechanical Turk. Their model is based on probabilistic graphical models invented specifically for this task.

The first approach using neural networks is proposed by Bisk et al. (2016b), who describe and compare several neural models for understanding natural language commands. Their dataset (Bisk et al., 2016a) contains simulated world with square blocks and actions descriptions in English (see Figure 1). Since the actions are always shifts of single block to some location, they divide the task into two: predicting which block should be moved and where. They call these tasks source and target predictions. With their best model, they reach 98% accuracy for source prediction and 0.98 average distance between correct and predicted location for target.

The world is represented by x and y coordinates of 20 blocks. Each block has a digit or logo of a company for easy identification. There are 16,767 commands in the dataset, divided into train, development, and test set. The commands were written by people using Amazon Mechanical Turk and therefore contains many typos and other errors.

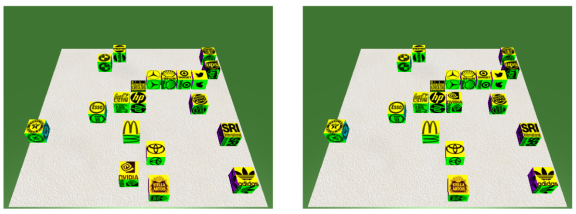


Figure 1: Visualisation of command "Move Nvidia block to the left of HP block" in our world.

In this paper, we propose several models solving this task and report improvement compared to the previous work by Bisk et al. (2016b).

2 Models

2.1 Data preprocessing

For tokenization of commands we use simple rule based system. Because of the typos we use Hunspell[1], which is a widely used spell checker. Finally to prevent overfitting of neural models we replace all tokens with less than 4 occurrences in training data with special token representing unknown word.

2.2 Benchmark model

To be able to measure the impact of the RNN based models, we first introduce a simple rule-based benchmark. The benchmark searches for block numbers (*1, 2, . . . , 20, one, two, . . . twenty*), logo names (*adidas, bmw, burger, king, . . .*),[2] and directions (*west, north, east, south, left, above, right, below*) in the commands.

[1] http://hunspell.github.io/

[2] If the logo name contains more words (e.g. *burger king*), this model search for each part of the word and for concatenation of both words (e.g. *burger, king*, and *burgerking*).

Proceedings of the First Workshop on Language Grounding for Robotics, pages 44–48,
Vancouver, Canada, July 30 - August 4, 2017.

For predicting the source (which block should be moved), the model predicts the block corresponding to the first word in the sentence denoting a block. For predicting the target location (where the source block should be moved), the model predicts position of the last word describing block If there exist words describing directions, the last one is chosen and the position is changed by one in the direction corresponding to the word.

For example, in the command

> *Put the **UPS** block in the same column as the **Texaco** block, and one row **below** the **Twitter** block.*

the benchmark model finds three words describing blocks (*UPS, Texaco*, and *Twitter*) and the word *below* describing direction. The block word (*UPS*) is predicted as source. As the target location, the banchmark model chooses the current location of *Twitter* block (the last block word) moved one tile down, because of the *below* word.

2.3 Neural model with world on the input

Our first neural model is relatively straightforward. Word embedding vectors representing the tokenized command are given to a bidirectional LSTM recurrent layer (Hochreiter and Schmidhuber, 1997). The last two states of both directions are concatenated together with the world representation (2 coordinates for each of the 20 blocks). and fed into single feed forward layer with linear activation function. For predicting source, this layer has dimension 20 and its outputs are then used as logits to determine the source block. For predicting location the last feed forward layer has dimension two and its outputs are directly interpreted as predicted target location.

2.4 Predicting reference and relative position

Our second model is similar to the one proposed by Bisk et al. (2016b).

It does not predict directly the target location, but a meaning representation of the command, which is then interpreted based on the world state to get the final predicted target location. Our representation is composed of 20 weights representing how much each block is used as a reference, and 2-dimensional vector representing the relative position from the reference block. Let $w = (w_1, w_2, ...w_{20})^T$ represent the weights of individual reference blocks, $d = (d_1, d_2)^T$ represent the relative position and

$$S = \begin{pmatrix} s_{1,1} & s_{1,2} & \dots & s_{1,20} \\ s_{2,1} & s_{2,2} & \dots & s_{2,20} \end{pmatrix}$$

be the state of world, where $s_{1,i}$ and $s_{2,i}$ are x and y coordinates of the i-th block The final target location $l \in R^2$ is then computed as $l = Sw + d$.

In most commands, the target is described in one of the following ways:

1. By reference and direction: *Move BMW above Adidas*
2. By reference, distance and direction: *Move BMW 3 spaces above Adidas*
3. By absolute target: *Move BMW to the middle of bottom edge of the table*
4. By direction relative to source: *Move BMW 3 spaces down*
5. By two references: *Move BMW between Adidas and UPS*

This representation is able to capture the meaning of all of these. For example, the command 1 can be represented as $w = (1, 0, 0, ..., 0)^T, d = (0, 1)^T$, the command 5 as $w = (0.5, 0, 0, ..., 0, 0, 0.5)^T, d = (0, 0)^T$.[3]

The tokenized one-hot encoded command is given to a bidirectional LSTM recurrent layer, the two last states are concatenated and fed into two parallel feed-forward layers. The first one has 20 dimensions and outputs the weights w of references, the second one is 2-dimensional and outputs the relative position d. The target location is then computed from these.

2.5 Using recurrent output layers

We also tested a variant of the previous architecture in which the feed-forward output layers are substituted by recurrent 128-dimensional LSTM layers. The new architecture is shown in Figure 2.

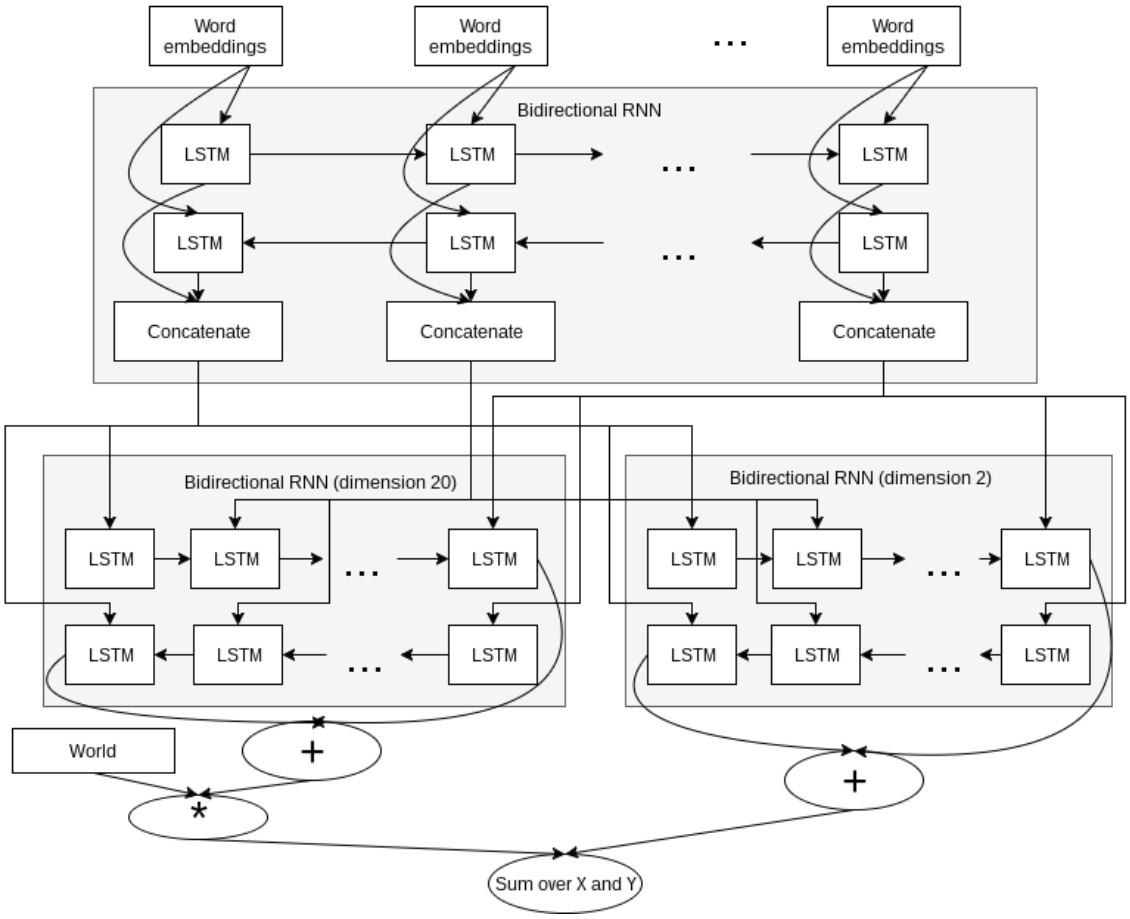
We also tried similar models for predicting the source blocks. They have bidirectional recurrent layer, followed by single output layer, which is feed-forward for one model and recurrent 64-dimensional LSTM for the other one.

3 Results

The experiment results are compared in Table 3. We report improvement over the previous results for both source and target location predictions. For source prediction the network without world

[3]The Adidas block has weight w_1 and the UPS block is the last with weight w_{20}.

Figure 2: Final target prediction network, which uses recurrent layer instead of feed-forward one.



on the input and with feed-forward output layer achieves accuracy 98.8%.This is better than the best model of Bisk et al. (2016b), who reported 98% accuracy. The improvement is mainly caused by preprocessing data with spell checker and better hyperparameter selection. Without using spell checker our model has accuracy 98.3%.

As for the target location prediction, our best model has average distance of 0.72 between predicted and correct target location. This is an improvement over both rule based benchmark with 1.54 and the best model reported by Bisk et al. (2016b), who had 0.98. The median distance is 0.04 which is much better than their comparable End-To-End model with median distance 0.53. In 65.8% of test instances the distance of our model is less than 0.5, which might be considered a distinctive line between good and bad prediction.

4 Error analysis and discussion

We manually analyzed bad predictions of our best model. As for the source block prediction, there were only 18 mistakes made on the devset:

1. The two-sentence command (7 mistakes). In the first sentence, it looks like the first mentioned block is the source, but the second sentence states otherwise.[4] *"The McDonald's tile should be to the right of the BMW tile. Move BMW."*
2. Block switching (3 mistakes): *"The 16 and 17 block moved down a little but switched places."*
3. Commands with typos (3 mistakes): *"Slide block the to the space above block 4"* (Note that the third word here should be *three*.)
4. Commands including a sequence (2 mistakes): *"Continue 13, 14, 15..."*
5. Grounding error (2 mistakes), see Table 1.
6. Annotation error (once, not a mistake).

Major improvement of source accuracy may be achieved by solving the problem where second sentence changes the meaning of the first one. However, there are no similar commands in the training data, so it is hard to come with solution.

[4] All these seven sentences were likely written by single author.

Mistake type	#	Description & Example
More reference blocks	31	Target location is described using two or more reference blocks. *Place block 12 on the same horizontal plane as block 9, and one column left of block 14.*
Source same as reference block	11	Model mistakes source for reference. Typically, the last block mentioned in the sentence is source. *Move block 10 above block 11, evenly aligned with 10 and slightly separated from the top edge of 10.*
Annotation error	11	Command does not make sense or does not describe the correct action. *Move Pepsi to the slight southwest until it's north of Pepsi.*
Missing reference block	9	No reference block in the command. *Move the Stella block down to the very bottom of the square.*
Large direction	9	Distance between reference and correct location is more than 1 block. *move the texaco block 5 block lengths above the BMW block*
Grounding error	8	Unusual description of blocks and typos in block names. *Put the block that looks like a taurus symbol just above the bird.*
Learning mistake	8	Relatively simple example, yet still bad prediction. *Block 4 should be moved almost straight down until it is resting on block 5.*
Others	13	

Table 1: The worst predictions analysis: Probable reasons behind bad predictions in 100 worst instances of the development set. *References* are the blocks which are used in the command for describing the target location.

	Source	Target
Random baseline	5.4%	6.12
Middle baseline	5.4%	3.46
Rule-based benchmark	96.3%	1.54
Bisk et al. (2016b)	98%	0.98
World as input	98.5%	3.05
Feed-forward output layer	98.8%	1.07
Recurrent output layer	98.5%	0.72

Table 2: Results comparison. In *random* and *middle* baselines, the randomly chosen block is placed on random position or in the middle of the board.

Similarly, the word *switch* appears only once in the training set.

Overall we think that for source prediction we reached the limitations given by the dataset we are using and without usage of another data it is very hard to get significant improvements.

For target prediction we divide 100 worst predictions into categories, which can be seen in Table 1.

11 out of the 100 worst predictions are bad because the commands does not make sense. But also in many other commands the target location is not described precisely, so the overall impact of inaccurate commands is in our opinion bigger and it also influences the training of models.

The other problem categories except of Learning mistake have similar underlying cause. The sentence structure is unusual and does not appear in the training data very often. Also in some cases such as the More references category the sentences are more complicated.

But even though these sentences are challenging and the model makes mistakes in them relatively often, it works well for majority of these sentences. Thus we find out that our proposed sentence representation is in practice capable of representing almost all sentences in the dataset.

5 Conclusion

We presented four different architectures of neural networks for solving the task of robot communication on dataset by Bisk et al. (2016a). Our last model surpassed the previous reported results and reached accuracy of 98.8% for source prediction and 0.72 average distance between predicted and correct target location. We find out that our model is capable of understanding wide variety of commands in natural language and make mistakes mostly in sentences with features, which are badly or not at all represented in the training data.

References

Yonatan Bisk, Daniel Marcu, and William Wong. 2016a. Towards a dataset for human computer communication via grounded language acquisition. In *Workshops at the Thirtieth AAAI Conference on Artificial Intelligence*.

Yonatan Bisk, Deniz Yuret, and Daniel Marcu. 2016b. Natural language communication with robots. In *Proceedings of NAACL-HLT*. pages 751–761.

Kais Dukes. 2013. Semantic annotation of robotic spatial commands. In *Language and Technology Conference (LTC)*.

Ting Han and David Schlangen. 2017. Grounding language by continuous observation of instruction following. *EACL 2017* page 491.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9(8):1735–1780.

Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. 2006. Walk the talk: connecting language, knowledge, and action in route instructions, proceedings of the 21st national conference on artificial intelligence.

Stefanie A Tellex, Thomas Fleming Kollar, Steven R Dickerson, Matthew R Walter, Ashis Banerjee, Seth Teller, and Nicholas Roy. 2011. Understanding natural language commands for robotic navigation and mobile manipulation .

Matthew R Walter, Matthew Antone, Ekapol Chuangsuwanich, Andrew Correa, Randall Davis, Luke Fletcher, Emilio Frazzoli, Yuli Friedman, James Glass, Jonathan P How, et al. 2015. A situationally aware voice-commandable robotic forklift working alongside people in unstructured outdoor environments. *Journal of Field Robotics* 32(4):590–628.