Automatic Translation of Spanish Natural Language Commands to Control Robot Comands based on LSTM neural network

Mtr. Ing. Félix David Suárez Bonilla
Escuela de Ingeniería Electrónica / Eléctrica
Universidad Técnica Nacional
San José, Costa Rica.
felixdavidsuarezbonilla@gmail.com

Dr. ret. nat. Federico Ruiz Ugalde
Instituto de Investigaciones en Ingeniería
Universidad de Costa Rica
San José, Costa Rica
fruiz@eie.ucr.ac.cr

Abstract— **In this paper, we propose a high level layer able to translate motion commands in natural spanish language to a formal intermediate representation called Robot Control Language (RCL). The layer was built by using the seq2seq TensorFlow library, with a single forward LSTM for encoder and decoder respectively. We were able to achieve a 4.3e-08 loss employing a manually generated corpus in Spanish.**

Keywords—robotic architecture; neural machine translation; object model system; interface layer; complex tasks; reasoning and planning layer; physical control layer.

I. Introduction

The future incorporation of robots in human activities will require them to comprehend orders expressed in natural language, react to the environment, recognize the surroundings and understand human behavior [1]. This communication between a robot and a human depends on how the robot recognizes high level tasks and communicates them to the control layer, in a robotic architecture [2] [3].

Understanding of motion commands is one of the abilities that a robot should be able to perform. For example, if a human tells a robot to move to the room by saying *"go to the end of the hall and enter the room"*, the robot should be able to translate this complex task into an intermediate formal representation, which can be finally translated to low-level control commands. The software and the hardware must be prepared to accomplish this translation.

In this paper, we adopt a robotic architecture with three layers: a high level layer meant to process complex tasks, the interface layer meant to translate atomic tasks into inputs to physical object controllers and a low level layer meant to perform physical object control. The low level layer contains an *object model system* which consists in mathematical models of the objects so we can predict and control their behavior. The object model is a concept intended to abstract the robot body from the high level layer [5] [6].

The object model must be able to perform a variety of actions, which are specified by the atomic tasks coming out from the high level layer. One action that has been implemented as an object model is *slide*, where a rigid planar body slides on a horizontal planar surface. This single action requires to find a solution for three double-integral equations over the surface of the object for every possible center of rotation [7]. The construction of object models is not a trivial problem.

The high level layer is usually in charge of translating orders and instructions expressed by a human to an intermediate formal representation, so we can avoid the ambiguity and the absence of syntax and semantic rules which is inherent to natural language [4]. This layer is also in charge of reasoning and planning. In this research, we focus on the natural language processing part of the problem, in order to generate a more transparent experience to the robot user.

This study is meant to propose a high level layer component able process commands in natural language with the purpose of generating an intermediate action language, a formal representation that can be used by downstream layers in a robotic architecture. The intermediate action language that was chosen is Robot Control Language (RCL), this is a motion oriented language made of route instructions through an indoor environment containing objects and landmarks [12].

In this paper, we use Neural Machine Translation (NMT) to implement a system that is able to translate Spanish to robot commands (RCL). In section II, we review the previous work made by Sutskever, Trinanda and Kahuttanaseth. In section III, we analyze a robotic architecture of three layers and we show how the language can be translated from the top to the bottom layer. In section IV, we study the neural network that will be used in this research. In section V, we explain how the system was implemented. In section VI, we show the results of the experiment. Finally, in section VII, we detailed the conclusions and the future work of this research.

II. Related work

In 2014, Sutskever and Vinyals proposed a recurrent neural network (RNN), based on a multilayered Long Short-Term Memory (LSTM) cell, to translate from English to French. Their main achievement was to present a general approach that shows better results compared to the conventional word by word translation algorithms used in NMT [34].

Trinanda (2015) followed a similar approach to translate Indonesian natural language to robot actions [4]. This system consisted in two parts: a speech recognition component and a natural language processing system. The speech recognition experiments got 82% of accuracy, and the natural language processing part obtained 93%.

A similar algorithm was also used by Mehreen (2017) for Roman-Urdu to Urdu transliteration, in which both, the language and the alphabet change. Their model was able to predict sentences up to length 10 while accomplishing a Bilingual Evaluation Understudy (BLEU) score of 48.6 on the testing dataset [44].

The most recent research, made by Kahuttanaseth (2018), also used Neural Machine Translation (RNN+LSTM) to translate from english motion commands, expressed in natural language, to a more formal representation meant to be understood by a robot. This system was implemented in Python and Tensorflow and achieved a 79.25% of accuracy [22].

There is something common regarding these previous works. The output language is still not a formal mobile robot language such as: ARCL (A Robot Control Language) [36], ZDRL (Zhe Da Robot Language) [37], and RCL (Robot Control Language). Also, according to the systematic literature review, no work has been made to translate motion robot orders in Spanish to an intermediate formal robot language by using NMT with LSTM neural networks; despite the fact that Spanish is one of the most used languages in the world.

III. Robotic architecture

A. Arquitecture Design

The assumed robotic architecture has been divided in three layers: a high level layer, an interface layer, and a low level layer. The high level layer processes "complex tasks" and splits it into more simple tasks which can be executed by the low level layer, these simple tasks are called "atomic tasks". The low level layer contains the controllers which are in charge of controlling the objects [7] [24] [25] [29].

The high level layer performs planning and reasoning; and the low level layer is in charge of prediction, perception and physical object control. We can find very different challenges in both layers; the high level layer is more related to: artificial intelligence, machine learning, natural processing language and knowledge representation. In the other hand, the low level layer is more related to: mechanics, automatic control, mathematical modeling and kinematics.

The interface layer in the middle maps the "atomic task" with the respective controller inputs in the low level layer. For example, taking the atomic task: *"Push the glass of milk away"*; the verb "push" would translate to an input domain region that uses a push controller, "glass of milk" specifies the object to be controlled, and "away" gives the desired outcome of the object [14] [15].

The high level layer performs the segmentation of "complex tasks" into more simple actions (atomic tasks). This layer has been addressed by Ruiken, Muller, Gorges and Ternorth [8] [9] [10] [11]. The implementations of this high level layer starts with observations of human activities and performs optimizations to find the atomic tasks which minimize energy and effort [29].

B. Object model system

The object model describes how the object reacts to certain physical inputs. The inputs to the object model are any physical quantities that could change the state of the object over time such as: force and torque. The output refers to physical values or properties of the object, such as: position, speed, acceleration and temperature. The object model system describes the internal workings of the object and it contains the controllers which are required to implement the respective actions which are contained within the atomic task [5] [6] [7] [23].

The main advantage of using object models as part of the low level layer is that, instead of concentrating on the robot motion, the attention is put on the objects to be manipulated. This helps to close the communication gap between humans and robots because human language usually involves actions manipulating objects and not human articulations moving the objects. For example, we do not say "move the hand to slide the box", we just say "slide the box". Human language is usually referred to objects and not to the human body [5] [6] [7] [23].

C. Language

In the context of this research, it is important to distinguish three types of languages: natural language, action language, and robot control commands [26] [27] [28] [31] [32]. The high level layer processes the natural language and generates an action language. The action language is processed by the interface layer to generate inputs to the physical object controllers. Finally, the low level layer generates the control commands for the robot articulation to execute the given tasks.

Reiterating, the natural language refers to any sentence expressed by a human that generally consists in complex tasks, such as:

- *Go to the end of the hall and turn right*
- *Go to the third junction and turn left*
- *Get out of the room and go to next junction*

The action language consists of atomic instructions pointed to the execution of a minimal and well-defined task, for example:

- *do-until (junction current-loc) (move-to forward-loc)*
- *turn-right*
- *do-until (not (room current-loc)) (move-to forward-loc)*

The robot commands correspond to the specific language used to operate the robot hardware directly. For example, two instructions in KUKA Robot Language® might look like below:

- LIN {X 17.3, Y 26.0, Z 55.0}
- PTP {X 12.3, Y 30.0, Z 50.0}

The LIN robot command moves the object or tool at a defined velocity along a straight path to the given position. The PTP instruction moves the object along the fastest path to the end point. The fastest path is not usually a straight line [33].

IV. Sequence to sequence models

Sequence-to-sequence (seq2seq) models are machine learning models that has proven to achieve remarkable performance on challenging problems such as: speech recognition, text summarization, visual object recognition, and neural machine translation (NMT).

In NMT, conventionally, translation was made by splitting sentences into multiple pieces and then translated them phrase by phrase. This approach has a big problem, the meaning of a sentence is determined by the whole sentence and not by portions or phrases of it. Seq2seq addresses this challenge by capturing long-range dependencies [39].

From a high level point of view, a sequence to sequence model is made of two main components: an encoder, and a decoder (See figure 1). The encoder is in charge of looking into the input sequence and encoding it into a vector called “context vector”. In the other side, the decoder is accountable for moving through the output while reading the context vector.

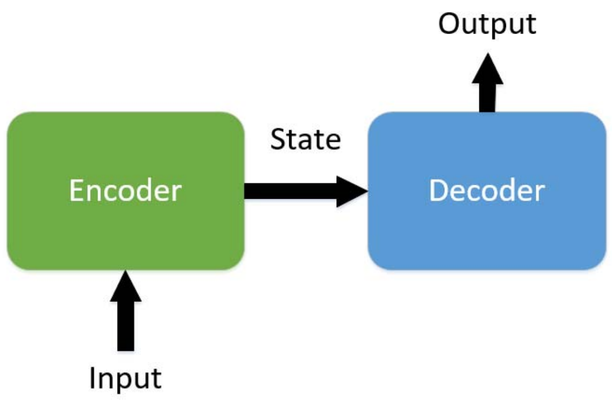


Fig. 1. Seq2seq model architecture that consists in two main components: an encoder and a decoder.

In the seq2seq arquitecture, the performance is considered to be low for long sequences. This occurs because of small internal representation. To overcome this challenge, an important feature called “attention” can be used. It works by providing an improved context and a learning algorithm that tells the decoder to focus over specific range of the sequence.

Before going deeper into this topic, we need to understand how a recurrent neural network (RNN) works.

A. Recurrent neural network

RNN are neural networks with loops in them, allowing information to persist. It is a neural network in which there is a hidden state and an output which operates on a variable-length input sequence [35]. At each time, the hidden state $\boldsymbol{h(t)}$ is updated by:

$$\boldsymbol{h}(t) = f(\boldsymbol{h}(t-1), \boldsymbol{x})$$

where f is a non-linear activation function and $\boldsymbol{x}$ is a variable-length sequence. The activation function can be as simple as a sigmoid function and as complex as a Long Short-Term Memory (LSTM) unit.

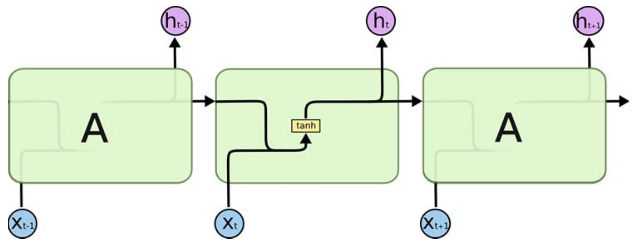


Fig. 2. Simple recurrent neural network that contains a very simple structure, one *tanh* layer [39].

B. Long short-term memory network

There is an exceptional kind of RNN called LSTM which is widely used in NMT. A common LSTM unit is composed of: a cell, an input gate, an output gate, and a forget gate. A LSTM network can remember values, but not only the previous value; it is able to remember values for either long or short periods of time. An LSTM is suited to classify, process, and predict time series.

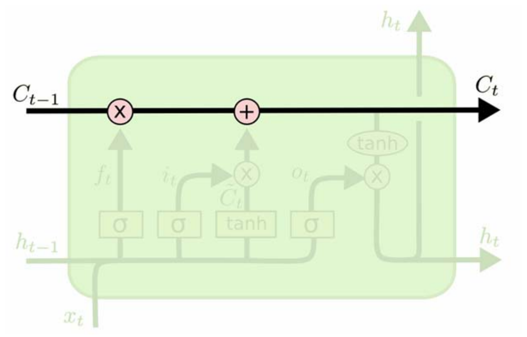


Fig. 3. LSTM unit with the cell state C_t being modified through the top of the diagram [39].

The core idea in LSTMs is the cell state; the horizontal line running through the top of the block diagram (See figure 3). The unit can remove or add information to the cell state, regulated by the gates.

The first element is the forget gate, which is composed of a sigmoid neural layer and a pointwise multiplication operation. This allows to multiply the state by numbers between zero and one, describing how much of the state is allowed to go through (See figure 4).

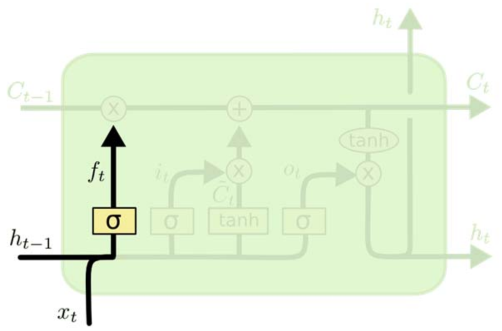


Fig. 4. LSTM unit with the forget layer being highlighted which is meant to decide what information we are going to throw away from the cell state [39].

The next step decides which information will be stored in the cell state. This layer has two parts: a sigmoid layer called the **input gate** and a tanh layer which creates a value that could be added to the state (See figure 5).

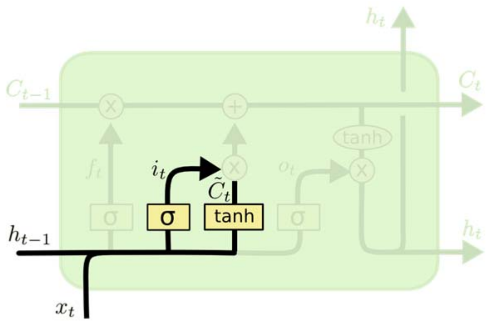


Fig. 5. LSTM unit where the cell is updated based on the input gate and the tanh layer values [39].

Finally, it is required to decide the output, which will be a filtered version of the calculated cell state. First, a sigmoid layer decides what parts of the cell state will be output. Then, the cell state goes through a tanh layer and it is multiplied by the sigmoid state (See figure 6).

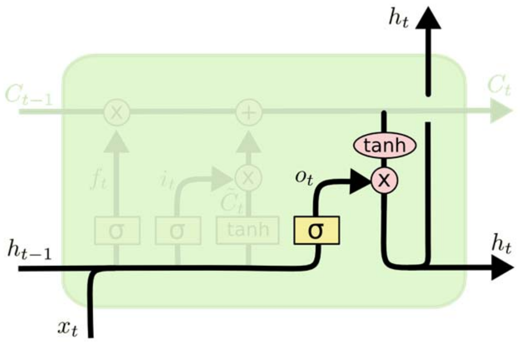


Fig. 6. LSTM unit where the output is calculated based on the cell state, the sigmoid layer and the tanh layer [39].

There are multiple variants of LSTM architectures, the one being described here is considered a pretty normal LSTM.

C. *Neural machine translation*

LSTM is the core element of NMT, since it is used in both main components: the encoder and the decoder (See figure 7). The encoder calculates a representation for each source sentence, based on that representation, the decoder generates a translation, one target word at a time.

An example is shown in figure 8 where the source sentence “I am a student” is translated into the target sentence “Je suis étudiant”. At a high level, the encoder analyzes the input source sentence without making any prediction; the decoder, on the other hand, processes the target sentence while predicting the next words.

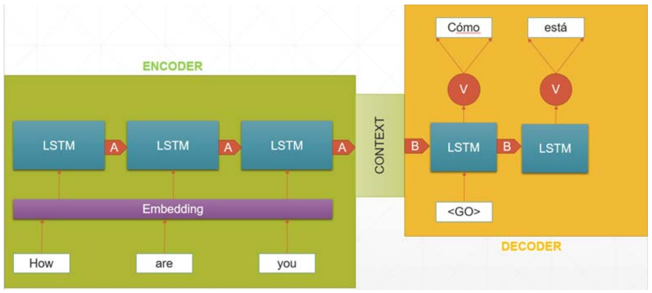


Fig. 7. LSTM is used in NMT, in this case the sentence “How are you” in english is translated to “Como está” in spanish.

In more detail, the encoder and the decoder get the source sentence, then a marker “-”, and the target sentence. Given these words, the models finds out the embeddings to generate the corresponding word representations. This embedding layer works with weights, one set per language, which are learned during training [40].

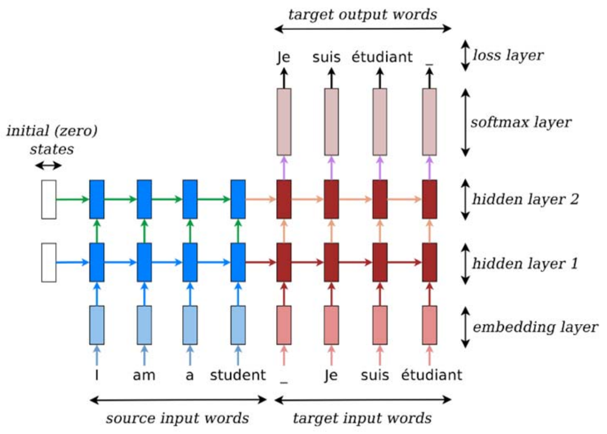


Fig. 8. NMT process to translate the source sentence “I am a student” to a target sentence “Je suis étudiant” [40].

The quality of neural machine translation can be measured by using the evaluation metric called BLEU. For this metric, quality is considered to be the correspondence between a machine's output and that of a human being. Scores are computed by comparing sentences with a set of good translations. Then, these values are averaged over the whole corpus to estimate the translation metric [45].

V. Implementation

The proposed software system was implemented in Python using Tensorflow, an open source library for machine learning developed by Google. The system is flexible and can be used in a wide variety of fields such as: speech recognition, computer vision, robotics, natural language processing, and geographic information extraction [38].

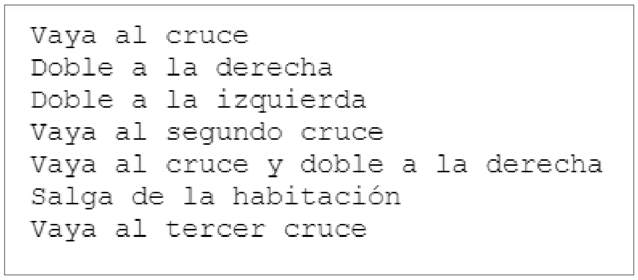
By employing the seq2seq library from TensorFlow, a single forward LSTM encoder to single LSTM decoder architecture was implemented. To improve the quality of the trained encoder, a pre-trained word2vector genism model was used [41]. As there are not any publically available robot language word2vector model, a simple embedding was created for the decoder word retrieval.

To generate the output word embedding, we employed genism library, an open source utility for vectorial space modeling [42]. Using a seven sentence robot language corpus, that was exhaustive enough to incorporate all robot language commands, a word2vector model was created and read back to tensor flow.

A. Training dataset

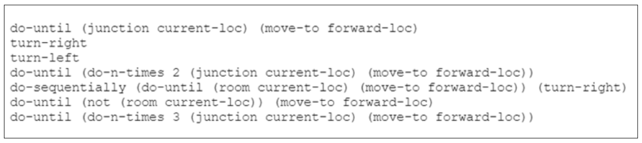
In order to train the neural network, we started with a small training dataset of seven motion commands in Spanish and their respective commands in RCL. The Spanish base corpus, made of seven lines or orders, is shown in figure 9. The corresponding RCL corpus, which were manually generated by the author, is shown in figure 10.

The seven motion commands were written by the researcher with the intention of being varied. They had to include the majority of control statements contained in RCL language and most of the common mobile instructions that are needed by a robot to go to a near place. For example: “turn right/left” is a key instruction if we want to indicate a robot where to go.



```
Vaya al cruce
Doble a la derecha
Doble a la izquierda
Vaya al segundo cruce
Vaya al cruce y doble a la derecha
Salga de la habitación
Vaya al tercer cruce
```

Fig. 9. Training Corpus made of seven motions commands in spanish natural language.



```
do-until (junction current-loc) (move-to forward-loc)
turn-right
turn-left
do-until (do-n-times 2 (junction current-loc) (move-to forward-loc))
do-sequentially (do-until (room current-loc) (move-to forward-loc)) (turn-right)
do-until (not (room current-loc)) (move-to forward-loc)
do-until (do-n-times 3 (junction current-loc) (move-to forward-loc))
```

Fig. 10. Training corpus made of seven motions orders in Robot Control Language which were generated by hand.

To increase this corpus we used a python program to substitute the Spanish words by their respective verbal conjugations and synonyms. By doing this, we generated a bigger dataset of 3785 lines in both languages: Spanish and RCL. It is important to note that whether synonyms try to maintain the essence of the sentence sometimes that is lost due to greedy corpus generation.

B. Model Generation

Subsequently, the model was trained using an Adam optimizer with gradient clipping for ten thousand iterations. To improve BLEU, encoder input was reversed. The training process employed a 16 word batch as an input, and registered the average loss as a function of the expected translation.

The designed software has the flexibility to implement traditional seq2seq decoder-encoder architecture [40], decoder with attention [43] and multiple bidirectional layers on the encoder. Such flexibility allows us to measure what is the best configuration to translate from Spanish to robot language. The following section provides more details about the experiment setup and further sections discuss our findings.

C. Experiment Setup

Regarding the neural network architecture to train the model, there are 3 levels of freedom which are simple decoder-encoder, attention based decoding and bidirectional encoding. Attention based encoding is expected to improve translation as it allows the mode to focus on the input Spanish word whereas bidirectional encoding comes to solve long memory dependencies.

As an input, we used a corpus of 3785 sentences in Spanish and RCL. That corpus remained static during the experiment and had an average 5.23 (1.88 deviation) words per Spanish translation and 11.72 (3.23 deviation) words on the robot language translation. It is important to notice that such corpus incorporated unrealistic Spanish sentences due to synonym substitution.

The loss during the model training was printed every 500th iteration in order to register the model speed between the different neural network configuration.

VI. RESULTS

The simple seq2seq model served as a baseline to understand if translating from Spanish to robot language was possible. The figure 11 shows the loss during the model training, and it’s important to realize that since the original 250 iteration, the model was able to predict with a low average error.

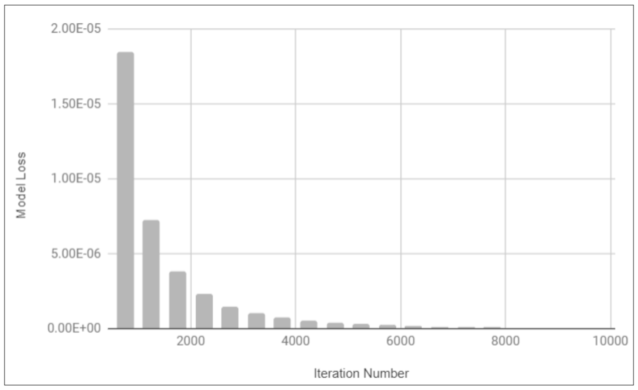


Fig. 11. Loss reduction during Adam Optimizer processing

Additionally, it is possible to see that the model almost reaches a perfect translation except for the ending parenthesis problem which can be seen in figure 12. The neural network is not able to understand when to stop translating. Traditional seq2seq model stop translating when an end sentence character is produced by the model, yet the parenthesis is such a common token of the robot language that its use is cumbersome.

```
.....................................................................................................
Step  10000
Actual: do-until ( not ( room current-loc ) ) ( move-to
forward-loc ) </s>

Predicted: do-until ( not ( room current-loc ) ( move-to
forward-loc ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) )
) ) ) ) ) ) ) ) ) ) ) ) ) )


Actual: do-until ( junction current-loc ) ( move-to forward-
loc ) </s>
Predicted: do-until ( junction current-loc ) ( move-to
forward-loc ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) )
) ) ) ) ) ) ) ) ) ) ) ) ) ) ) )

============= Step  10000  =============
            Loss:  2.629707528926035e-08
```

Fig. 12. Loss reduction during Adam Optimizer processing

In order to further improve the accuracy of the model, and fix the end of translation problem, attention decoder and bidirectional translation where tried. The table 1 registers the results of this approach. Attention model degraded the performance of the model which could be linked to the short sentence lengths used in the translation. Adding a bidirectional layer did not yield better performance, nor did it fix the end of sentence identification problem.

Table 1. Loss changes with differente neural network architectures

Architecture	Loss
Simple Seq2Se1	2.83e-08
Attention Decoder	0.08
Bidirectional Encoder	3.18e-08

By using different architectures for the neural network, we were not able to train the model to understand when a translation completed. Additional training rounds of above architectures did not yield better results due to the limited size of the training corpus.

VII. Conclusions

One challenge, present in this area, is absence of dictionaries for complex and atomic tasks in natural language. This absence is not absolute, Tenorth and Beetz addresed this problem by collecting and processing kitchen recipes in the English language [16]. We considered that addressing this problem can lead to more research in this area.

With an artificial corpus however, it can be seen that neural network translation using seq2seq models can produce an acceptable translation. We consider that miscorrelation between expected and predicted translation can be easily self-corrected by a compiler.

The presented proposal only analyzed the translation capabilities of robot movement, yet incorporating this learnings to additional robot capabilities is trivial. Our findings reflect that the advancements in spoken language translation can be reused in other nontraditional language processing.

The use of seq2seq models is recommended for very long corpuses. For future work, we recommend employing at least a ten thousand word corpus which is what traditionally is used. Additionally, the use of beam search translation has improved Google's neural machine translation which could be easily incorporated to our proposed model.

VIII. Acknowledgements

The present paper is a partial result of a research project entitled "Manipulation of everyday objects using an object model system. Creation of an object model library and a new object model, 322-B6-279" funded by the Research Agency of the University of Costa Rica (Vicerrectoría de Investigación). Additional funding and support is provided by the University of Costa Rica Dean's office, Electrical Engineering department, Engineering Research Institute, and National Technical University.

References

[1] M. A. V. J. Muthugala and A. G. B. P. Jayasekara, "A Review of Service Robots Coping With Uncertain Information in Natural Language Instructions," in IEEE Access, vol. 6, pp. 12913-12928, 2018.

[2] F. Ruiz-Ugalde, G. Cheng and M. Beetz, "Fast adaptation for effect-aware pushing," 2011 11th IEEE-RAS International Conference on Humanoid Robots, Bled, 2011, pp. 614-621.

[3] S. Cousins, "Challenges of building personal robots" 2011 IEEE Hot Chips 23 Symposium (HCS), Stanford, CA, 2011, pp. 1-36.

[4] K. T. Putra, D. Purwanto and R. Mardiyanto, "Indonesian natural voice command for robotic applications", 2015 International Conference on Electrical Engineering and Informatics (ICEEI), Denpasar, 2015, pp. 638-643.

[5] Ruiz-Ugalde, F., G. Cheng and M. Beetz: Prediction of action outcomes using an object model. In 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1708–1713, October 2010.

[6] Ruiz-Ugalde, F., G. Cheng y M. Beetz: Fast adaptation for effect-aware pushing. In 2011 11th IEEE-RAS International Conference on Humanoid Robots, pp. 614–621, October 2011.

[7] Ruiz Ugalde, Federico: Compact Models of Objects for Skilled Manipulation, 2014. Thesis.

[8] Ruiken, D., T. Q. Liu, T. Takahashi y R. A. Grupen: Reconfigurable tasks in belief-space planning. In 2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids), pp. 1257–1263, November 2016.

[9] Muller, Armin, Ra Kirsch and Michael Beetz: Transformational planning for everyday activity. In 17th international, pp. 248–255, 2007.

[10] Gorges, N., A. J. Schmid, D. Osswald y H. Worn: A framework for creating, coordinating, and executing skills on a humanoid robot. In 2007 7th IEEE-RAS International Conference on Humanoid Robots, pp. 385–391, November 2007.

[11] Tenorth, M. and M. Beetz: Priming transformational planning with observations of human activities. Universidad Técnica de Munich, Intelligent Autonomous Systems, Garching, Germany, 2010.

[12] Matuszek, C. Learning to Parse Natural Language Commands to a Robot Control System. International Symposium on Experimental Robotics (ISER) , 2012.

[13] Thenmozhi, D., R. Seshathiri, K. Revanth and B. Ruban: Robotic simulation using natural language commands. In 2017 International Conference on Computer, Communication and Signal Processing (ICCCSP), pp. 1–4, Jan 2017.

[14] Domingos, Pedro: What's missing in AI: The interface layer. University

of Washington, Washington, USA, 2007.

[15] Domingos, Pedro y Daniel Lowd: Markov Logic: An Interface Layer for Artificial Intelligence. Morgan and Claypool Publishers, 1st edition, 2009, ISBN 1598296922, 9781598296921.

[16] M. Tenorth, D. Nyga and M. Beetz, "Understanding and executing instructions for everyday manipulation tasks from the World Wide Web," 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, 2010, pp. 1486-1491.

[17] Yamazaki, K., Y. Watanabe, K. Nagahama, K. Okada and M. Inaba: Recognition and manipulation integration for a daily assistive robot working on kitchen environments. In 2010 IEEE International Conference on Robotics and Biomimetics, pp. 196–201, Dec 2010.

[18] Winkler, J. and M. Beetz: Robot action plans that form and maintain expectations. In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5174–5180, Sept 2015.

[19] Lagrand, C., M. v. d. Meer and A. Visser: The Roasted Tomato Challenge for a Humanoid Robot. In 2016 International Conference on Autonomous Robot Systems and Competitions (ICARSC), pp. 341–346, May 2016.

[20] Blodow, N., L. C. Goron, Z. C. Marton, D. Pangercic, T. Ruhr, M. Tenorth and M. Beetz: Autonomous semantic mapping for robots performing everyday manipulation tasks in kitchen environments. In 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 4263–4270, Sept 2011.

[21] Feix, T., I. M. Bullock y A. M. Dollar: *Analysis of Human Grasping Behavior: Correlating Tasks, Objects and Grasps.* IEEE Transactions on Haptics, 7(4):430-441, Octubre 2014, ISSN 1939-1412.

[22] W. Kahuttanaseth, A. Dressler and C. Netramai, "Commanding mobile robot movement based on natural language processing with RNN encoder-decoder", 2018 5th International Conference on Business and Industrial Research (ICBIR), Bangkok, Thailand, 2018, pp. 161-166.

[23] Krivic, S., E. Ugur y J. Piater: A robust pushing skill for object delivery between obstacles. In 2016 IEEE International Conference on Automation Science and Engineering (CASE), pp. 1184–1189, Agosto 2016.

[24] Quack, B., F. Worgotter y A. Agostini: Simultaneously learning at different levels of abstraction. In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4600–4607, September 2015.

[25] Zhang, C. and J. A. Shah: Co-optimizing task and motion planning. In 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4750–4756, October 2016.

[26] Ferrein, Alexander, Christopher Maier, Clemens Muehlbacher, Tim Niemueller, Gerald Steinbauer and Stavros Vassos: Controlling Logistics Robots with the Action-based Language YAGI. In Proc. of 9th International Conference on Intelligent Robotics and Applications (ICIRA2016), Tokyo, Japan, 2016.

[27] Guerra-Filho, G. y Y. Aloimonos: A Language for Human Action. Computer, 40(5):42–51, Mayo 2007, ISSN 0018-9162.}

[28] Sugiura, Komei and Naoto Iwahashi: Learning Object-manipulation Verbs for Human-robot Communication. In Proceedings of the 2007 Workshop on Multimodal Interfaces in Semantic Interaction, WMISI '07, pp. 32–38, New York, NY, USA, 2007. ACM, ISBN 978-1-59593-869-5.

[29] Fang, J., J. Zhao, F. He and X. Lin: Design and research of three-layers open architecture model for industrial robot software system. In 2013 IEEE International Conference on Mechatronics and Automation, p´aginas 104–109, August 2013.

[30] Calli, B., A. Walsman, A. Singh, S. Srinivasa, P. Abbeel and A.M. Dollar: Benchmarking in Manipulation Research: Using the Yale-CMUBerkeley Object and Model Set. IEEE Robotics and Automation Magazine, 22(3):36 – 52, 2015.

[31] Zhang, Shiqi, Fangkai Yang, Piyush Khandelwal and Peter Stone: Mobile Robot Planning Using Action Language, pp. 502–516. Springer International Publishing, Cham, 2015, ISBN 978-3-319-23264-5.

[32] Glenberg, Arthur M. and Michael P. Kaschak: Grounding language in action. Psychonomic Bulletin & Review, 9(3):558–565, Sep 2002, ISSN 1531-5320.

[33] I. Koštál, "A .NET application searching for data in a log file of the KUKA industrial welding robot". Proceedings of the 16th International Conference on Mechatronics - Mechatronika 2014, Brno, 2014, pp. 656-661.

[34] Sutskever, I. and Vinyals, O. Sequence to Sequence Learning with Neural Networks. Neural Information Processing Systems Conference, 2014. Neural Information Processing Systems Foundation.

[35] Cho, Kyunghyun. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pp. 1724-1734, October 25-29, 2014, Doha, Qatar.

[36] A. S. Elmaghraby, "A robot control language," Southeastcon '88., IEEE Conference Proceedings, Knoxville, TN, 1988, pp. 413-416.

[37] Baokang Chen and Vuliang Xu, "ZDRL: A Motion-Oriented Robot Programming Language," Proceedings of the 1988 IEEE International Conference on Systems, Man, and Cybernetics, 1988, pp. 1226-1229.

[38] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, and Zhifeng Chen. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[39] Olah, Christopher. Understanding LSTM Networks. Blog. Posted on August 27, 2015. URL: https://goo.gl/8MN7MP

[40] Luong, Minh-Thang. Neural Machine Translation. A dissertation submitted to the department of computer science and the committee on graduate studies on Stanford University, December 2016.

[41] UCHILE-NLP. uchile-nlp/spanish-word-embeddings. GitHub [online]. [Accessed 4 July 2018]. Available from: https://goo.gl/dXUfci

[42] Python Software foundation. Gensim Library. Website. Visited on July, 2018 . URL: https://pypi.org/project/gensim/

[43] VASWANI, Ashish, et al. Attention is all you need. En Advances in Neural Information Processing Systems. 2017. p. 5998-6008.

[44] M. Alam and S. ul Hussain, "Sequence to sequence networks for Roman-Urdu to Urdu transliteration," 2017 International Multi-topic Conference (INMIC), Lahore, 2017, pp. 1-7.

[45] M. Yang et al., "Extending BLEU Evaluation Method with Linguistic Weight," 2008 The 9th International Conference for Young Computer Scientists, Hunan, 2008, pp. 1683-1688.