Learning Robot Structure and Pose Embeddings using Graph Neural Networks

J. Taery Kim [1] Jeongeun Park [2] Sungjoon Choi [2] Sehoon Ha [1]

Abstract

We propose a learning framework to find the representation of a robot's kinematic structure and pose embedding spaces using graph neural networks (GNN). Finding a compact and low-dimensional embedding space for complex phenomena is a key for understanding its behaviors, which may lead to a better learning performance, as we observed in other domains of images or languages. However, although numerous applications deal with various types of structural and motion data, the embedding of the generated data has been relatively less studied by roboticists. To this end, our work aims to learn embeddings for two types of robotic data: the robot's design structure, such as links, joints, and their relationships, and the pose data, such as kinematic joint positions. Our method exploits the tree structure of the robot to train appropriate embeddings to the given robot data. To avoid overfitting, we formulate multi-task learning to find a general representation of the embedding spaces. We evaluate the proposed learning method on a robot with a simple linear structure and visualize the learned embeddings using t-SNE. We also study a few design choices of the learning framework, such as network architectures and message passing schemes.

1. Introduction

Computation robot design has been discussed in terms of motion planning (Ha et al., 2017) or control learning (Zhao et al., 2020), but researchers haven't investigated the nature of the morphological and motion data thoroughly. Given the fact that the robot's structure can be represented as a graph, it seems to be a natural choice to leverage graph neural networks (GNNs) to represent the robot's data. Particularly, we employ the framework of message passing neural networks (MPNNs) proposed by Gilmer et al. (Gilmer et al., 2017) as the network architecture. This scheme has been applied to learn a variety of graph-structured data, from a single large graph (*e.g.*, social network) to multiple small graphs (*e.g.*, molecules). While the former data typically involves the per-node or per-edge data, the latter focuses on graph-level tasks. Inspired by the success of molecule works (Wu et al., 2018), we also formulate the learning with graph-level tasks.

[1]Georgia Institute of Technology, Atlanta, GA 30308, USA
[2]Department of Artificial Intelligence, Korea University, Seoul, Korea.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

In addition, we take an approach of multi-task learning (MTL) when we learn embeddings to find a shared representation that is agnostic to the selection of tasks. In particular, we set two related tasks in this work: solving forward kinematics (FK) and inverse kinematics (IK) while sharing robot structure representation. Because the robot's morphologies and motions are related to each other, we expect that this simultaneous learning of FK and IK allows us to find general embeddings more efficiently. Note that our goal differs from other existing tasks (Reily et al., 2020), which do not consider any kinematic constraints.

We evaluate the proposed learning framework on a simple 2D robot with a linear structure. We generate a set of data while varying the number of joints and link lengths, and augment the structure data set with pose data. Then we process the given high dimensional data to find a compact fixed eight-dimensional embedding space. We visualize the learned embedding space using t-SNE and compare it with the embedding of the multilayer perception (MLP) to find insights into the robot design and motion spaces.

2. Learning Robot Embedding with GNN

Our method encodes robotic data into two separated spaces: structure embedding and pose embedding. The pose embedding is decoupled from the structure embedding, so that structure embedding targets particularly for invariant information which is specific to the robot structure. Instead, the pose embedding helps to capture kinematics information, which involves robots' physical feasibility, robotic properties, or skills expected to learn through the embeddings.

2.1. Tree Message Passing

We can model a robot's structure as a tree by converting joints to nodes and links to edges while assuming no loops as shown in Fig. 1a. Then a leaf node represents a null joint that makes the connected edge an end-effector. We focus

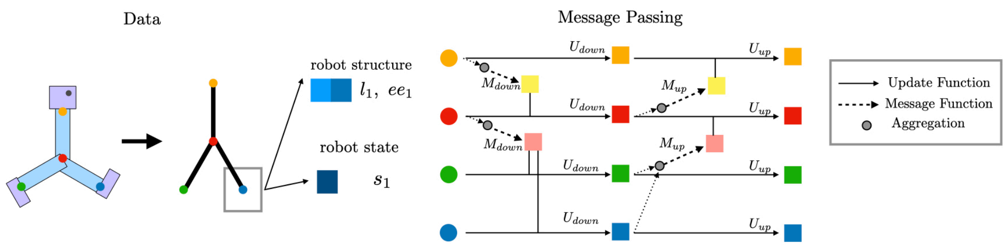

Figure 1. Tree message passing overview. (a) Robot structure converted to tree-structured data with nodes and edges. Robotic data such as joint angle s and link length l are assigned to node or edge features. (b) Two paths in message passing: *downward passing* to spread the information from parent to child nodes, then *upward passing* to aggregate the information at the root node. M, A, and U denote the message generation, message aggregation, and hidden state update function, respectively, for each direction $down$ and up.

on this tree structure of the robots and exploit it to learn compact but meaningful structural or pose embeddings.

Fig. 1b describes how the messages are generated and passed throughout the structure. Each message passing step updates the hidden state h_r of receiver node r with message function M, aggregation function A, and update function U. Let us assume a set of n sender nodes $\mathcal{S} \in [s_1, s_2, \cdots, s_n]$ that send their message to the receiver node r. Then each node $s \in \mathcal{S}$ will generate its own message using the function M:

$$m_s^t = M_{dir}(h_s^t, e_{(s,r)}), \tag{1}$$

where dir denotes the direction of the message passing, and $e_{s,r}$ denotes the feature of the edge (s, r). Either parent or child node can be the sender based on the direction of the message passing. Then, the messages from the senders are aggregated by the aggregation function A_{dir} as

$$m^t = A_{dir}(m_{s_1}^t, m_{s_2}^t, \cdots, m_{s_n}^t). \tag{2}$$

Finally, the hidden state of the receiver node r is updated as

$$h_r^t = U_{dir}(h_r^{t-1}, m^t) \tag{3}$$

where t denotes the time step of the update.

Within the tree structure, we define message passing in two directions, from root to leaf or leaf to root, which will be referred to as *downward passing* and *upward passing*, respectively. For the downward passing, each node is updated only based on its parent node, thus an identity function is used as an aggregation function A. This passing can also be viewed as spreading a parent node's information to child nodes. Contrarily, upward message passing has to collect the information from multiple child nodes using an aggregation function. Typical choices of an aggregation function include mean, pooling, and recurrent neural networks.

2.2. Pretraining of Encoders

Ahead of multi-task learning, we pre-train both structure and pose encoders to facilitate the whole learning process. This is because we observe that the proposed multi-task learning is difficult to learn from scratch. For this purpose, we pre-train structural encoders by learning new self-reconstructing tasks that encode node features and decode to reconstruct the given inputs. We suggest two reconstruction strategies using tree message passing: encoding-decoding (ED), which reconstructs all the nodes at once, and fill-in-the-blank (FB), reconstructing randomly masked nodes.

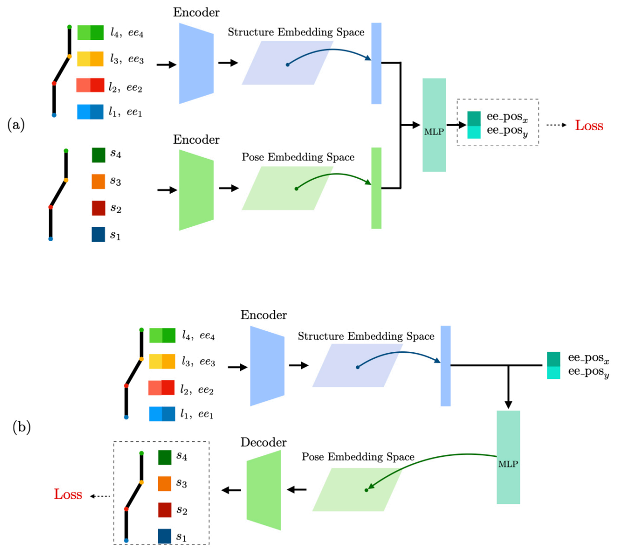

Figure 2. Two tasks solved jointly as multi-task learning: (a) forward kinematics, FK; and (b) inverse kinematics, IK. The tasks share structure embedding space and pose embedding space while training. FK predicts the end-effector position from the structure and pose embeddings, while IK estimates the pose from the structure embedding and end-effector position.

2.3. Multi-task Representation Learning on Kinematics

We suggest learning an embedding space of robot through multi-task representation learning (MTL) as illustrated in Fig. 2 to learn the space that is generalizable across multiple tasks. Because our goal is to make the embedding space of the robot's kinematics, we select two popular tasks: forward kinematics and inverse kinematics. For the structure

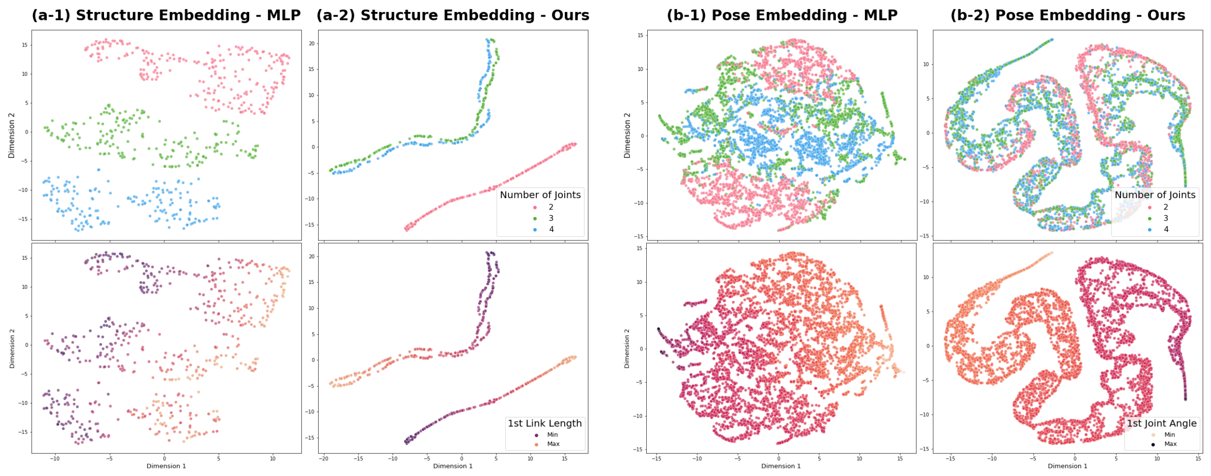

Figure 3. Embedding space visualizations using t-SNE: (a) Structure embedding (b) Pose embedding. We plot the same embedding with two different color schemes, where they denote the number of joints in the upper row and the first link length in the bottom row.

embedding, we use link lengths l and binary flags ee to note whether the node is an end-effector as node features. For the pose embedding, we use joint positions s as the node features, which vary in a single robot structure.

The first task, forward kinematics (FK, Fig. 2a), is to find the end-effector position in Cartesian space from the given joint angles. First, we obtain a structure embedding of a robot using a structure-tree encoder while computing a pose embedding from joint angles using a pose-tree encoder. Then, both the embeddings are concatenated and fed to a multilayer perceptron to predict the end-effector position.

On the other hand, the inverse kinematics task (IK, Fig. 2b) predicts joint angles from the given position of the end-effector. Note that the IK task naturally has ambiguity with multiple solutions while the FK task predicts the unique end-effector position. The structure embedding is obtained through the aforementioned structure-tree encoder and then combined with the given end-effector position. Then an MLP layer maps the combined information to the pose-tree embedding space to reconstruct the pose-tree structure, which holds the joint angle configuration.

We learn on a linear combination of multi-tasks losses:

$$\mathcal{L}_{total} = w_{\text{FK}}\mathcal{L}_{\text{FK}} + w_{\text{IK}}\mathcal{L}_{\text{IK}}. \tag{4}$$

where w_{FK} and w_{IK} are set as 5.0 and 0.5, respectively.

3. Experiments

In this pilot study, we focused on a simple robot with a linear structure. We built our data set using the Reacher environment in OpenAI Gym (Brockman et al., 2016), which is a well-known benchmark in reinforcement learning. A reacher is a robot arm with one end-effector, so we can represent the robot's structure with a tree with one child node for each parent node. We vary the number of joints and the length of each link to generate various robot structures while changing the poses of each robot to learn kinematics. The structure and pose encoders are trained to encode the given input into a fixed-dimension vector, which is empirically set to 8. The size of hidden layers in MLP is also set to 8.

3.1. Embedding Space Visualization

To explore the learned latent space, we plot the embedding vectors to two-dimensional space using t-SNE (Van der Maaten & Hinton, 2008) as shown in Fig. 3. The color in the upper row indicates the number of joints, while the bottom row figures are colored based on the first node's feature value, *i.e.*, link length or joint angle.

Structure embedding. Fig. 3a compares the structure embedding spaces learned with two architectures, our GNN with fill-in-blank architecture (column (a-1)) and the standard MLP (column (a-2)). We present both embeddings with two different colorizations based on the number of joints (upper) and the link length of the first node (lower). Both embeddings build clusters based on those two features, the number of joints and the link length of the first node, which affect the kinematic performance of the robot significantly, i.e., the reachable workspace of the robot. However, the overall shapes are quite different from each other: the MLP produces more dispersed clusters while our method creates two long-curved lines. We suspect that our method maintains a single cluster for three-linked and four-linked structures because both offer multiple solutions for a single IK problem.

We argue that one embedding is not particularly better than the other. Rather, they just provide different perspectives for understanding the robot structure data. The MLP embedding space is not what we wanted to obtain because it does not give us any intuition for relating structures with different numbers of links. On the other hand, the embedding

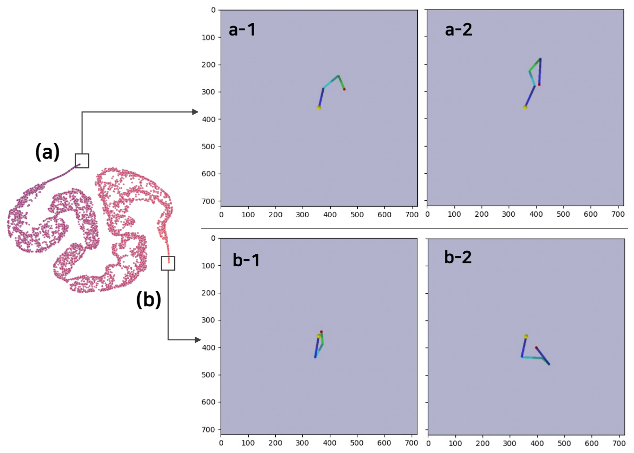

Figure 4. Sampled robot poses from pose embedding space. (a) and (b) are each end of the cluster. The pose embeddings are mostly determined by the first joint angle.

with our approach associates structures with three and four links, which can be reasonable in terms of their kinematic capabilities, such as reachability and singularity against IK.

Pose embedding As shown in Fig. 3b, pose embeddings are in a single cluster and correlate to the node feature (i.e., joint angle), not the number of joints. The embeddings of MLP (col. b-1) are distributed all over the space without any recognizable shape, but ours are connected in a lengthy cluster with sharp ends (col. b-2). Fig. 4 shows the actual robot poses sampled from each end of the pose embedding space learned with our method. Left column robots have three joints, while right column robots have four joints. Similar to our observation from the latent space visualization, the embedding is mostly correlated to the first joint angle that determines the range of end-effector position.

3.2. Reconstruction Strategies Comparison

In Fig. 5, we further compare two different reconstruction strategies discussed in Sec. 2.2. ED reconstruction embeddings (Fig. 5a) create a long line connected all together with a bit of noise, but no specific correlation shown with neither the number of node nor node features. Contrarily, the embeddings learned by FB reconstruction are in more stretched shapes but subject to the node features. That is, FB reconstruction distinguishes each input in detail and be expected to facilitate further tuning in the embedding space.

4. Conclusion

In this paper, we have proposed a tree-based message passing neural network for creating an embedding space for robotic structures and poses. We leverage the tree structure existing in kinematic structures to encode the given robot data into the corresponding embedding spaces. We further incorporate the kinematic movements of the robot through the multi-task learning to learn more meaningful representations. We visualized the learned embeddings via t-SNE

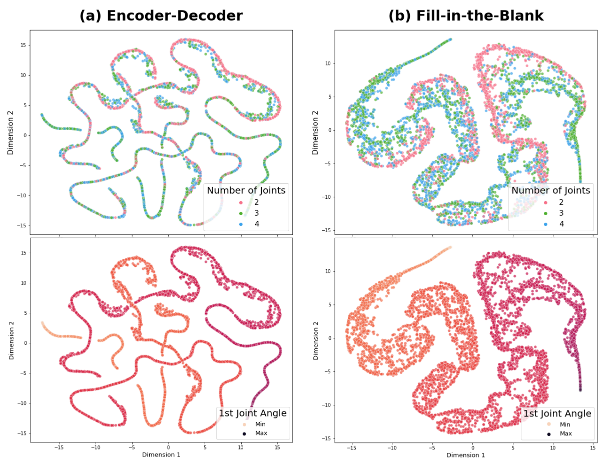

Figure 5. Comparison of reconstruction strategies. (a) Encoder-decoder (ED) (b) Fill-in-the-blank (FB).

dimensional reduction and analyzed how the embedding space formed for different structures and pose data. In the future, we hope to add more complexity to the robot structure, such as more nodes, various types of revolute joints, and graph structures including closed chain.

References

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *CoRR*, abs/1606.01540, 2016.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *Proc. of the International conference on machine learning*, pp. 1263–1272. PMLR, 2017.

Ha, S., Coros, S., Alspach, A., Kim, J., and Yamane, K. Joint optimization of robot design and motion parameters using the implicit function theorem. In *Robotics: Science and systems*, volume 8, 2017.

Reily, B., Reardon, C., and Zhang, H. Representing multi-robot structure through multimodal graph embedding for the selection of robot teams. In *Proc. of the International Conference on Robotics and Automation*. IEEE, 2020.

Van der Maaten, L. and Hinton, G. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., Leswing, K., and Pande, V. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

Zhao, A., Xu, J., Konaković-Luković, M., Hughes, J., Spielberg, A., Rus, D., and Matusik, W. Robogrammar: graph grammar for terrain-optimized robot design. *ACM Transactions on Graphics (TOG)*, 39(6):1–16, 2020.