IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received May 11, 2021, accepted May 23, 2021, date of publication May 28, 2021, date of current version June 9, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3084599

Pose-Graph Neural Network Classifier for Global Optimality Prediction in 2D SLAM

RANA AZZAM[1], FELIX H. KONG[2], TAREK TAHA[3], AND YAHYA ZWEIRI[1,4]

[1]KU Center for Autonomous Robotic Systems (KUCARS), Khalifa University of Science and Technology, Abu Dhabi 127788, United Arab Emirates
[2]Center for Autonomous Systems, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW 2006, Australia
[3]Robotics Laboratory, Dubai Future Foundation, United Arab Emirates
[4]Faculty of Science, Engineering and Computing, Kingston University London, London SW15 3DW, U.K.

Corresponding author: Rana Azzam (rana.azzam@ku.ac.ae)

This work was supported by the Khalifa University of Science and Technology under Award CIRA-2020-082 and Award RC1-2018-KUCARS.

ABSTRACT The ability to decide if a solution to a pose-graph problem is globally optimal is of high significance for safety-critical applications. Converging to a local-minimum may result in severe estimation errors along the estimated trajectory. In this paper, we propose a graph neural network based on a novel implementation of a graph convolutional-like layer, called PoseConv, to perform classification of pose-graphs as optimal or sub-optimal. The operation of PoseConv required incorporating a new node feature, referred to as cost, to hold the information that the nodes will communicate. A training and testing dataset was generated based on publicly available bench-marking pose-graphs. The neural classifier is then trained and extensively tested on several subsets of the pose-graph samples in the dataset. Testing results have proven the model's capability to perform classification with $92 - 98\%$ accuracy, for the different partitions of the training and testing dataset. In addition, the model was able to generalize to previously unseen variants of pose-graphs in the training dataset. Our method trades a small amount of accuracy for a large improvement in processing time. This makes it faster than other existing methods by up-to three orders of magnitude, which could be of paramount importance when using computationally-limited robots overseen by human operators.

INDEX TERMS Pose graph optimization, global optimality, graph neural network, simultaneous localization and mapping.

I. INTRODUCTION

Simultaneous localization and mapping (SLAM) is the problem of concurrently estimating a robot trajectory and a map of its surroundings while navigating in an environment. It has been thoroughly studied for more than three decades and employed in a wide variety of applications, some of which are very sensitive to the accuracy of the estimated map and trajectory. SLAM estimates are generated by minimizing the negative log-likelihood of a set of measurements that a robot obtains from the environment. This estimation takes place in the SLAM back-end where an algorithm is employed to resolve the constraints that are generated based upon the collected measurements. Graph SLAM [1] is one of the most common algorithms used to resolve the SLAM problem. Robot poses along the trajectory and the observable landmarks in the environment under investigation are represented as vertices in the graph. In case landmarks are not inserted into the map and only robot poses are considered, the algorithm is referred to as pose-graph optimization [2], [3]. Spatial constraints, that are formulated based upon the sensory measurements collected by the robot, are encoded as edges that connect the graph vertices. These measurements are susceptible to a wide range of uncertainties [4], including sensor noise and systematic biases. Consequent to such uncertainties, obtaining a perfect estimation of the robot trajectory is deemed impossible [5]. Hence, an inference technique is carried out to find the best configuration of robot poses and map landmarks that yields minimum errors when imposing the constraints [5]. A solution to a graph SLAM problem is referred to as globally consistent when the optimization outcome conforms to the true robot trajectory and the topology of the environment [6]. In fact, convergence to the globally optimal solution to a

The associate editor coordinating the review of this manuscript and approving it for publication was Huaqing Li.

SLAM problem is not guaranteed by the widely prevalent iterative optimization-based SLAM solvers [3], [7]. To that end, researchers have started to investigate the nature of the SLAM problem to profoundly understand its structure back in 2010 [8]. For a comprehensive survey that studies the properties of SLAM from a theoretical point of view, readers are referred to [9]. Several research works followed to determine the conditions under which certifying or achieving global optimality would be possible as will be discussed in the next section.

While existing global optimality certification methods (and indeed, solvers) for pose-graph SLAM can be impressively fast [10]–[12], in some applications where computational power is limited, additional speed may be desirable. Motivated by this, we propose a neural network architecture that can learn to distinguish between (globally) optimal and sub-optimal solutions of a 2D pose-graph SLAM problem. The proposed classification is carried out after obtaining a candidate solution of the pose-graph optimization problem in the SLAM back-end.

The proposed method is a fast, approximate global optimality estimator. The main advantage is speed, as our method is significantly faster than existing methods [10], [13]. The trade-off is that of accuracy, i.e. the classifier is not guaranteed to correctly identify a candidate pose-graph solution as optimal or sub-optimal. While this is a significant drawback, it allows robots with limited computational resources to perform an expedient optimality estimate, which might have been unattainable using existing methods. Additionally, many robots are currently overseen by human operators, who can be alerted by the classifier, and then visually assess the quality of the map to decide whether to continue operation.

At the heart of the proposed method is a neural network which predicts whether a candidate pose-graph solution is optimal or sub-optimal. Because of the non-Euclidean nature of the pose-graph data, we choose to adopt a graph neural network [14], [15] that can seamlessly operate on such data structure. In general, a graph is a data structure that exhibits a high level of expressiveness. It consists of vertices that are connected to each other via edges that represent certain relationships [16].

Our approach operates on pose-graphs that have nodes representing the 2D poses of a robot and edges that represent measurements obtained by the robot while transitioning between two connected nodes, and their corresponding measurement uncertainty. It is worth noting that our approach requires the translational and rotational measurement covariances to be spherical. We also propose to add a new node feature that will hold the information exchanged between the nodes while the proposed convolution-like operation, that we call PoseConv, operates on the pose-graph. More particularly, PoseConv implements an operation to compute the costs of all the pose-graph edges and aggregates this information in the *cost* feature of each node. The neural network then performs classification based on the cost features across the pose-graph and consequently verifies the pose-graph's global optimality. The other node and edge features remain unchanged. To achieve model generalization, classification is done based on the cost values of the graph edges rather than the node and edge features directly. To train the proposed model, we generate a dataset that contains several pose-graphs with different sizes and various noise model parameters. Then, we label pose-graph samples in the training dataset as optimal or sub-optimal by comparing the cost of each sample with the optimal cost associated with that sample (obtained by solving the dual semi-definite relaxation from [10]).

The aim of this study it to enable prompt global optimality verification of pose-graph estimates with high accuracy, which is approached through the following contributions:

- A novel operation on pose-graphs, termed PoseConv, is proposed to convey messages that facilitate the optimality verification of a 2D pose-graph.
- A supplementary node feature, referred to as *cost* is proposed to accommodate the messages exchanged among the nodes. This feature will hold information about the cost of a sub-graph in the node's neighborhood.
- A model that can learn to classify candidate solutions of 2D pose-graphs as optimal or sub-optimal, is developed based on the novel PoseConv layer.
- Extensive testing of the proposed model has been conducted on benchmark datasets and the model's generality and applicability to unseen data have been verified.

The rest of this paper is organized as follows. Section II reviews related research work from the literature. Section III presents in detail our proposed implementation of PoseConv, the architecture of the pose-graph neural classifier, and the training and testing dataset. In section IV, we evaluate the performance of our proposed approach through several tests on different portions of the training and testing datasets. Finally, the conclusions drawn from this work and possible future work directions are presented in Section V.

II. RELATED WORK

A. SLAM GLOBAL OPTIMALITY

Further to the theoretical analysis of the nature of SLAM presented in [8], several research studies were conducted to explore convergence guarantees. The work presented in [17] addresses the case when map joining is used to solve feature-based SLAM problems formulated as pose-graphs and studies the conditions that affect the number of existing local minima. In [18], the region of attraction of the global solution to the pose-graph optimization problem using Gauss-Newton was estimated. The work presented in [19] and [20] suggest that when strong duality holds, the optimal cost of a pose-graph optimization problem in a robotics application can be obtained by means of semi-definite programming. Empirical tests have shown that the duality gap is zero when the noise is below a critical threshold, which is commonly the case in practice [20]. Lagrangian duality was also employed in [21] to circumvent the non-convexity of the optimization constraints for the problem of rotation averaging and hence provide bounds on the optimal solutions to the problem, when noise levels are not critically high. Moreover, the work in [13] proposed a technique to verify if a given solution to a pose-graph is globally optimal. To accomplish that, the pose-graph optimization problem is first reformulated where chordal distance [22] is used instead of angular distance to represent the magnitude of the difference between two rotation matrices in $SO(2)$. Thereby, the optimization problem becomes amenable to implementing duality theory (e.g. [23]), using which bounds on the optimal cost of a pose-graph optimization can be computed. More specifically, upper and lower bounds on the optimal cost can be obtained using semi-definite programming. If the cost of a solution to the same pose-graph exceeds that bound, it is considered a (non-global) local minimum.

Another line of research was directed towards developing certifiably correct algorithms that guarantee global optimality of the pose-graph optimization estimate. SE-Sync [24] employs semi-definite programming to solve the convex semi-definite relaxation of the SLAM maximum likelihood estimation. Under practical noise regimes, SE-Sync is guaranteed to certify the correctness of its solution to the pose-graph optimization at hand. The same semi-definite relaxation was used in [25], along with a distributed optimization algorithm to certify the optimality of distributed pose-graph optimization.

B. GRAPH NEURAL NETWORKS

Graph neural networks (GNNs) are emerging deep learning approaches that have gained immense popularity in the past few years. GNNs were developed to target data structures that are not supported by the standard deep learning methods, like recurrent neural networks (RNNs) and convolutional neural networks (CNNs). The operation of RNN and CNN depends on the order in which the nodes appear in the input graph. Nevertheless, operations on graphs should depend on the connectivity of the nodes irrespective of their order. There are various types of GNNs that handle graphs in different ways. For comprehensive details on those, interested readers are referred to [14]. Our proposed approach adopts a message-passing GNN, whose operation happens in two stages; message passing and aggregation [16].

In recent years, GNNs have experienced expeditious proliferation and have proven their worth in a multitude of different applications [26], such as social network user localization [27], classification of images [28], aerial images [29], videos [30], text [31], and point clouds [32].

The structure of the employed graphs varies across the applications, where vertices and edges hold features that are meaningful to the problem in question. The closest graph structure to ours is that used to represent point-clouds in [32], where each node holds three features that denote the 3D coordinates of a point. Based on node features, edge features are then computed, to encapsulate the relationship between a node and its neighbors. A stack of EdgeConv [32] layers is then used to perform classification and segmentation of point-clouds. Each layer in the proposed model operates on the graph data and modifies the node and edge features accordingly.

III. PROPOSED APPROACH

In this paper, the problem of verifying the global optimality of a 2D pose-graph SLAM is addressed. A novel neural network classifier is trained to predict if a proposed solution to a pose-graph is optimal. Figure 1 illustrates the overall approach that is used to perform the prediction. In Section III-A, a novel implementation of a graph convolutional-like layer, referred to as PoseConv, is presented. This convolutional-like layer is used along with a stack of other fully connected layers to build a model that can learn to identify optimal solutions to pose-graphs. The detailed architecture of the model is described in Section III-B. The model was trained and tested using a dataset composed of variants of several publicly available pose-graphs as will be described in Section III-C.

A. POSE CONVOLUTION (POSECONV) LAYER

Consider a 2D pose-graph with n nodes denoted as $X = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}\}$ and m edges denoted as $E = \{e_1, e_2, \ldots, e_m\}$. Each node has four features; $\mathbf{x_i} = (x_i, y_i, \theta_i, cost_i)$. The first three features represent the robot pose. We propose to add a fourth feature, $cost_i$, to accommodate the messages that will be exchanged between the nodes in the neural network, as will be explained later. Edges represent relative measurements between nodes and are directional. An edge that connects $\mathbf{x_i}$ to $\mathbf{x_j}$ is denoted as e_{ij} and holds six features; $e_{ij} = (\Delta x_{ij}, \Delta y_{ij}, \Delta\theta_{ij}, \Omega_{\Delta x_{ij}}, \Omega_{\Delta y_{ij}}, \Omega_{\Delta\theta_{ij}})$. The former three features represent the relative position and orientation between $\mathbf{x_i}$ and $\mathbf{x_j}$, and the latter three are the diagonal entries of the information matrix.

The PoseConv operation on one node is defined by applying the sum aggregation function on the messages received along all the inbound edges. This operation only modifies the cost features of the graph nodes. The other node features and the edge features remain unchanged. A message from $\mathbf{x_i}$ to $\mathbf{x_j}$ is computed based on the features of $\mathbf{x_i}$, $\mathbf{x_j}$, and e_{ij} as shown in (1):

$$msg_{i,j} = (\alpha((\frac{u_x + u_y}{2} \times \left\| p_j - p_i - R_i\Delta_{ij} \right\|^2)) \\ +(\beta(u_\theta \times \frac{1}{2} \left\| R_iR_{ij} - R_j \right\|_F^2)) \quad (1)$$

where α and β are learnable parameters. $u_x = \sqrt{\Omega_{\Delta x_{ij}}}$, $u_y = \sqrt{\Omega_{\Delta y_{ij}}}$, and $u_\theta = \sqrt{\Omega_{\Delta\theta_{ij}}}$ represent the uncertainties of the measured Δx_{ij}, Δy_{ij}, and $\Delta\theta_{ij}$ respectively. $\|\mathbf{A}\|_F$ is the Frobenius norm of matrix $\mathbf{A}$. R_{ij} is the measured relative orientation between R_i and R_j where $R_i \in SO(2)$. Δ_{ij} is the measured relative position between p_i and p_j where $p_i \in \mathbb{R}^2$.

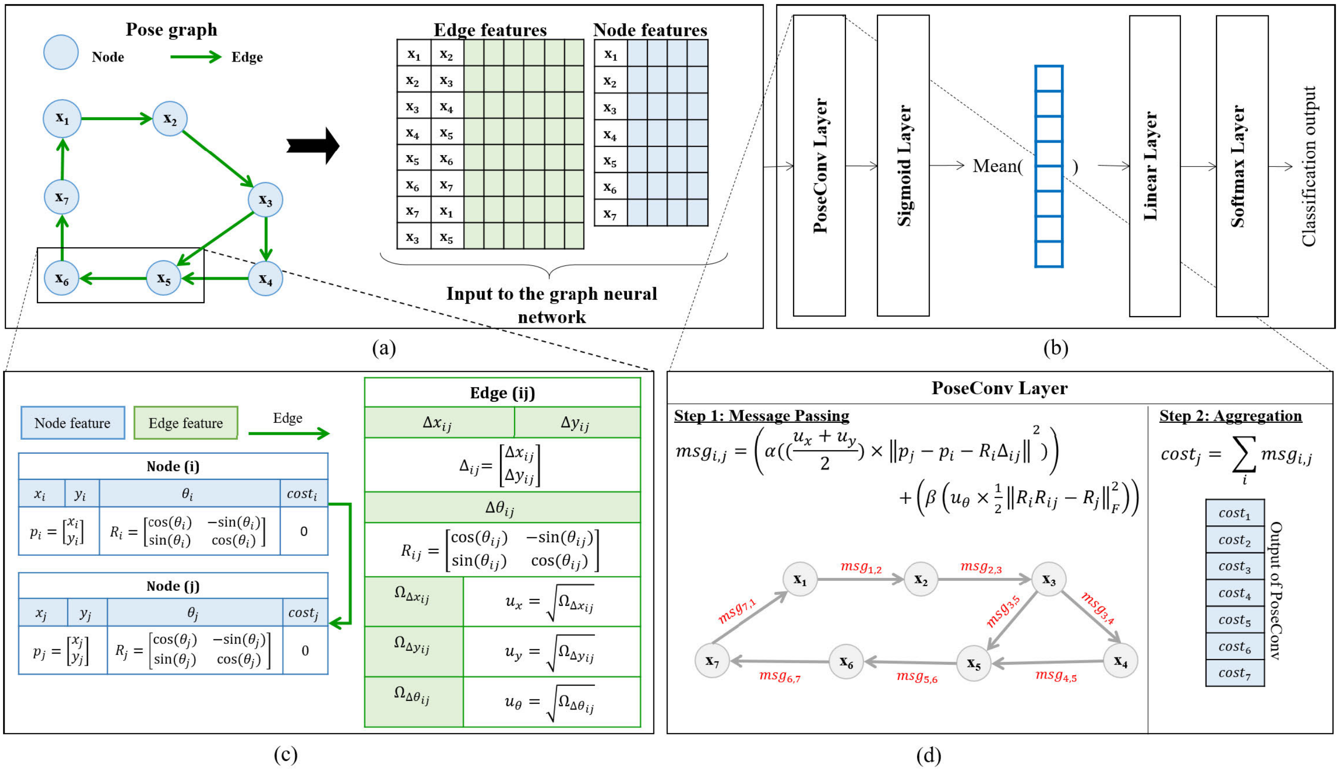

Pose graph
Node
Edge
x1 x2 x3 x4 x5 x6 x7
Edge features
Node features
Input to the graph neural network
(a)
PoseConv Layer
Sigmoid Layer
Mean( )
Linear Layer
Softmax Layer
Classification output
(b)
Node feature
Edge feature
Edge
Node (i)
x_i y_i θ_i $cost_i$
$p_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$ $R_i = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix}$ 0
Node (j)
x_j y_j θ_j $cost_j$
$p_j = \begin{bmatrix} x_j \\ y_j \end{bmatrix}$ $R_j = \begin{bmatrix} \cos(\theta_j) & -\sin(\theta_j) \\ \sin(\theta_j) & \cos(\theta_j) \end{bmatrix}$ 0
Edge (ij)
Δx_{ij} Δy_{ij}
$\Delta_{ij} = \begin{bmatrix} \Delta x_{ij} \\ \Delta y_{ij} \end{bmatrix}$
$\Delta\theta_{ij}$
$R_{ij} = \begin{bmatrix} \cos(\theta_{ij}) & -\sin(\theta_{ij}) \\ \sin(\theta_{ij}) & \cos(\theta_{ij}) \end{bmatrix}$
$\Omega_{\Delta x_{ij}}$ $u_x = \sqrt{\Omega_{\Delta x_{ij}}}$
$\Omega_{\Delta y_{ij}}$ $u_y = \sqrt{\Omega_{\Delta y_{ij}}}$
$\Omega_{\Delta \theta_{ij}}$ $u_\theta = \sqrt{\Omega_{\Delta \theta_{ij}}}$
(c)
PoseConv Layer
Step 1: Message Passing
$msg_{i,j} = \left(\alpha\left(\left(\frac{u_x + u_y}{2}\right) \times \|p_j - p_i - R_i\Delta_{ij}\|^2\right)\right) + \left(\beta\left(u_\theta \times \frac{1}{2}\|R_iR_{ij} - R_j\|_F^2\right)\right)$
Step 2: Aggregation
$cost_j = \sum_i msg_{i,j}$
$msg_{1,2}$ $msg_{2,3}$ $msg_{3,5}$ $msg_{3,4}$ $msg_{4,5}$ $msg_{5,6}$ $msg_{6,7}$ $msg_{7,1}$
$cost_1$ $cost_2$ $cost_3$ $cost_4$ $cost_5$ $cost_6$ $cost_7$
Output of PoseConv
(d)

FIGURE 1. Proposed approach.

The cost feature of node $\mathbf{x_j}$ is then computed as in (2).

$$cost_j = \sum_i msg_{i,j} \qquad (2)$$

where $msg_{i,j}$ is the message sent by node $\mathbf{x_i}$ that has an edge with node $\mathbf{x_j}$.

Figure 1(a) depicts the pose-graph structure, Figure 1(c) details the node and edge features, and Figure 1(d) shows the computations that are carried out in the PoseConv layer. The proposed PoseConv layer is stacked together with other fully connected layers to perform classification of pose-graphs as will be discussed in the next section.

B. PROPOSED NEURAL NETWORK ARCHITECTURE

Our contribution is a neural network classifier that attempts to classify a candidate pose-graph solution as optimal or sub-optimal. The proposed neural network architecture is depicted in Figure 1(b). The neural network accepts pose-graphs as input and does not require any pre-processing of its nodes or edge features. The cost feature for all the graph nodes is initially set to zero. The pose-graph, to be classified, is first passed to a PoseConv layer where the nodes exchange messages as described in Section III-A. Every node in the graph will aggregate the messages it receives from its neighbors using the sum aggregation function. The cost feature of every node in the graph is then updated to the outcome of the aggregation function. The output of the PoseConv layer is passed to a sigmoid layer, where the cost features will be updated accordingly. The mean of the resulting cost features is then computed and passed to a linear function, which will perform classification based on the cost features. This function will output a $k \times 2$ tensor, where k is the number samples in the input dataset and 2 is the number of classes: optimal or sub-optimal. The $k \times 2$ output tensor of the linear function is then passed to a softmax layer, which applies the softmax function shown in (3) to rescale the elements of the 2×1 tensor corresponding to every graph to the range [0, 1], while ensuring that they sum up to 1.

$$\mathbf{Softmax}(w_i) = \frac{e^{w_i}}{\sum_{j=1:2} e^{w_j}} \qquad (3)$$

The Adaptive moments (Adam) optimizer [33] with a learning rate of 0.01 is used to train the neural network by minimizing the cross entropy loss function.

C. TRAINING, VALIDATION, AND TESTING DATASETS

The dataset used to train, validate, and test the proposed neural network is generated based on publicly available, bench-marking pose-graphs including INTEL [34], FRH, FR079 [34], CSAIL [34], KITTI05 [35], KITTI06 [35], KITTI07 [35], and KITTI09 [35]. The pose-graphs have different sizes and were originally recorded in various environments. The measurements in each dataset were used to generate a set of *variants* of that dataset by adding extra additive noise to translational measurements, rotational

TABLE 1. Structure of the generated dataset.

Dataset	#nodes	#edges	$\sigma_{dx}(m)$	$\sigma_{dy}(m)$	$\sigma_{d\theta}(rad)$	#optimal samples	Total # samples
Intel	1728	2512	0-0.3	0-0.3	0-0.1	365	843
FRH	1316	2820	0-0.3	0-0.3	0-0.1	268	1028
FR079	989	1217	0-0.3	0-0.3	0-0.1	730	929
CSAIL	1054	1172	0-0.3	0-0.3	0-0.1	673	991
KITTI05	2761	2826	0-0.3	0-0.3	0-0.03	300	1131
KITTI06	1101	1150	0-0.3	0-0.3	0	741	771
KITTI07	1101	1106	0-0.3	0-0.3	0-0.06	437	1165
KITTI09	1591	1592	0-0.3	0-0.3	0-0.03	337	1331
					Total	3851	8189

measurements, or both. The additive noise parameters were incrementally varied between the minimum and maximum values reported in Table 1 for each dataset. The total number of variants for each pose-graph dataset is also reported in the same table, along with the corresponding number of optimal samples. The dataset contains 8189 pose-graph samples, 3851 samples of which are optimal. Hence, our developed dataset is balanced, with 47% of the samples being optimal and 53% being sub-optimal.

The optimal cost attainable for each pose-graph variant, which will herein after be referred to as the 'SDP value', was computed using SE-Sync [10], [24], [36]–[38]. More particularly, a convex semi-definite relaxation of the SLAM maximum likelihood estimation is first formulated. A specialized optimization algorithm is then employed to solve this semi-definite relaxation, while exploiting the graph structure of the problem. This optimization algorithm was used instead of other SDP solvers due to its efficiency and ability to certify the global optimality of a pose-graph solution. More details on the mathematical formulation of the problem can be found in [24].

Every pose-graph variant was also solved using the Levenberg-Marquardt (LM) optimizer. Then, each solution was evaluated by means of a cost function [13] that uses the chordal distance as a parameterization of the distance between two rotation matrices in $SO(2)$ as in (4). The chordal distance is defined as the Frobenius norm of the difference between the rotation matrices as seen in the second term of the following equation.

$$f = \sum_{ij}(||p_j - p_i - R_i\Delta_{ij}||^2 + \frac{1}{2}||R_iR_{ij} - R_j||_F^2) \qquad (4)$$

where $||A||_F$ is the Frobenius norm of matrix A, R_{ij} is the measured relative orientation between R_i and R_j, and Δ_{ij} is the measured relative position between p_i and p_j, corresponding to nodes i and j respectively. The definitions of the terms in (4) can be found in Figure 1.

Then, candidate solutions to the pose-graph problems that are generated by the LM solver are labeled as sub-optimal if their costs exceed the SDP value, and optimal otherwise. Figure 2 depicts a sample of the pose-graph variants that constitute the dataset.

IV. PERFORMANCE EVALUATION

The proposed pose-graph neural classifier was extensively tested with various partitions of the training and testing sets to verify its validity, applicability, and generality, as will be described here. Section IV-A presents the results of the leave-set-out cross-validation, where the generalization capability of the model will be illustrated. The results of the 10-fold cross-validation technique are then discussed in Section IV-B. Finally, the performance of the proposed neural network was compared to other architectures with a different number of PoseConv layers and the selection of our neural network architecture is justified.

A. LEAVE-SET-OUT CROSS-VALIDATION

This technique will evaluate how the proposed pose-graph neural classifier will generalize to unseen subsets of the dataset while training. As mentioned earlier, the dataset consists of variants of eight different pose-graphs: INTEL, CSAIL, FR079, FRH, KITTI05, KITTI06, KITTI07, and KITTI09. The dataset was partitioned into eight subsets, each containing only the variants of one of these pose-graphs. Then, the pose-graph neural classifier was trained and validated on each of these subsets (see Figure 3). For a fair judgement, the test was repeated using several initial random seeds. The mean and standard deviation of the resulting test accuracies across the different initial random seeds when each of the sets listed earlier was used as a test set are plotted in Figure 4.

It is evident that the model is capable of achieving high test accuracies for the various test sets. It is worth mentioning that the classifier was trained for 200 epochs only. In some cases, the tested model converges after few tens of training epochs where it achieves >92% accuracy. In other cases, though, convergence happens at a much lower rate, where after 200 training epochs the model achieves $80 - 90\%$ accuracy. We observed that changing the initial random seed affects convergence rate, but allowing more training epochs will allow the accuracy of every classifier in each subset to reach the accuracy of the highest-performing classifier, independent of seed. We discuss this in more detail in Section IV-C.

Hence for each test set, we have displayed the results where the initial random seed resulted in the model achieving

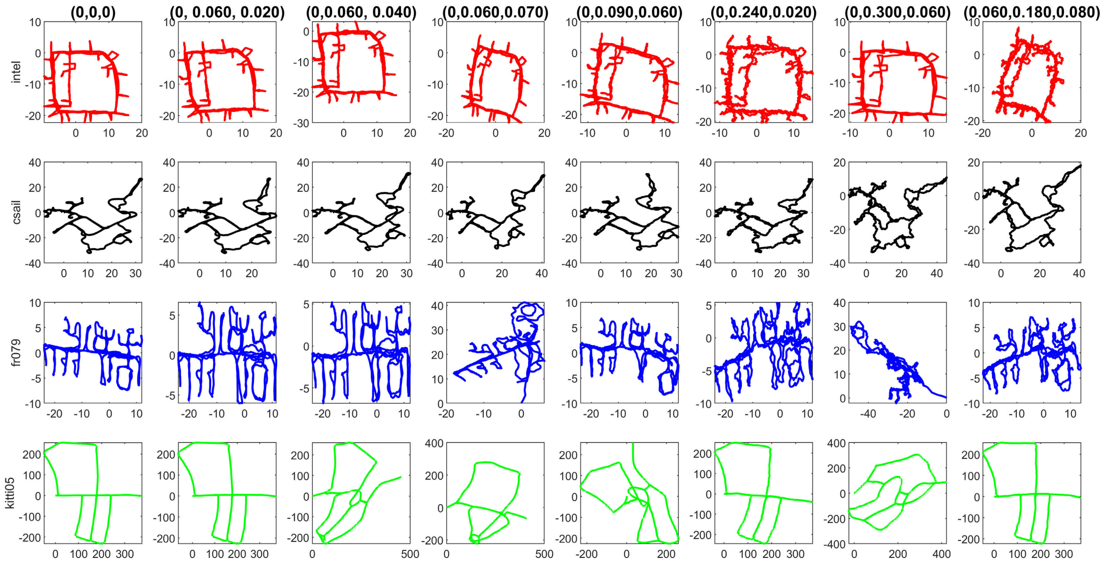
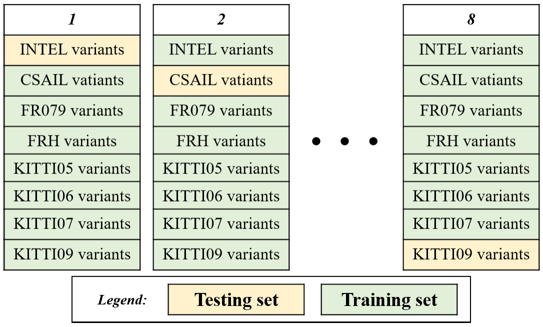


FIGURE 2. Sample pose-graphs from the generated dataset. Rows show variants of different datasets; INTEL, CSAIL, FR079, and KITTI05 respectively, and columns show different noise parameters. The title of each column in the figure indicates the noise parameters in the following format: (additive noise in relative orientation, additive noise in relative x position, additive noise in relative y position).



1	*2*		*8*
INTEL variants	INTEL variants		INTEL variants
CSAIL vatiants	CSAIL vatiants		CSAIL vatiants
FR079 variants	FR079 variants		FR079 variants
FRH variants	FRH variants	• • •	FRH variants
KITTI05 variants	KITTI05 variants		KITTI05 variants
KITTI06 variants	KITTI06 variants		KITTI06 variants
KITTI07 variants	KITTI07 variants		KITTI07 variants
KITTI09 variants	KITTI09 variants		KITTI09 variants

Legend: Testing set | Training set

FIGURE 3. Leave-set-out cross-validation technique.

the highest test accuracy. Figure 5 depicts the training and testing results for the selected models and Figure 6 shows the corresponding loss curves.

Figure 7 depicts sample testing results from the INTEL dataset when the INTEL pose-graph variants were used as a testing set and the remaining pose-graph variants were used for training. The figures show samples that are correctly labeled as optimal, correctly labeled as sub-optimal, incorrectly labeled as optimal, and incorrectly labeled as sub-optimal.

Overfitting is one of the most serious challenges that must be addressed when developing and training neural networks. It happens when the neural network does not have the capability to map unseen input data to the correct output [39]. In other words, the neural network fails to generalize well to data outside the training set. Several factors might cause the network to overfit, such as insufficient training data and high model complexity. The leave-set-out cross-validation test has

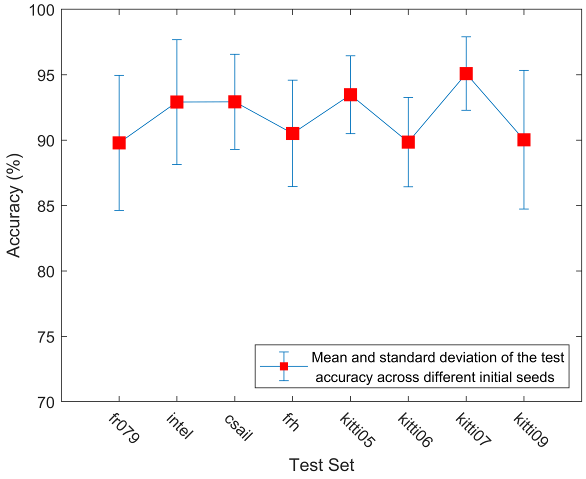


FIGURE 4. Leave-set-out cross-validation results with different initial random seeds.

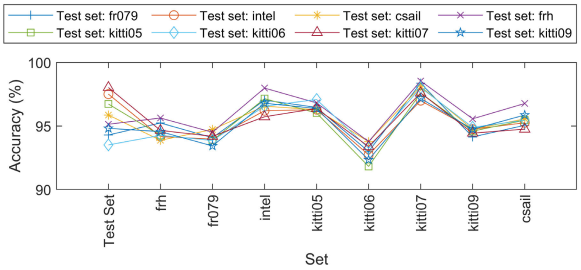


FIGURE 5. Leave-set-out cross-validation results.

demonstrated the generalizability of the proposed approach, where high accuracies were obtained when testing the neural network on variants of pose-graphs that were not present in

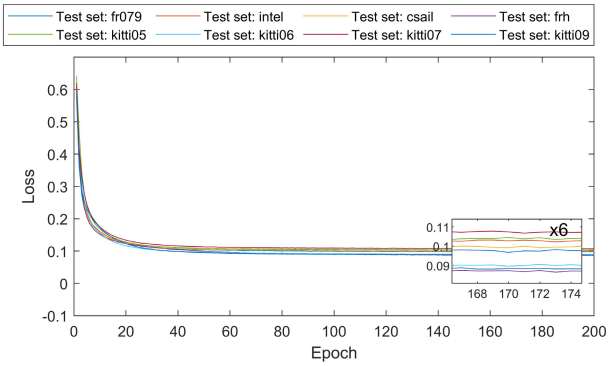
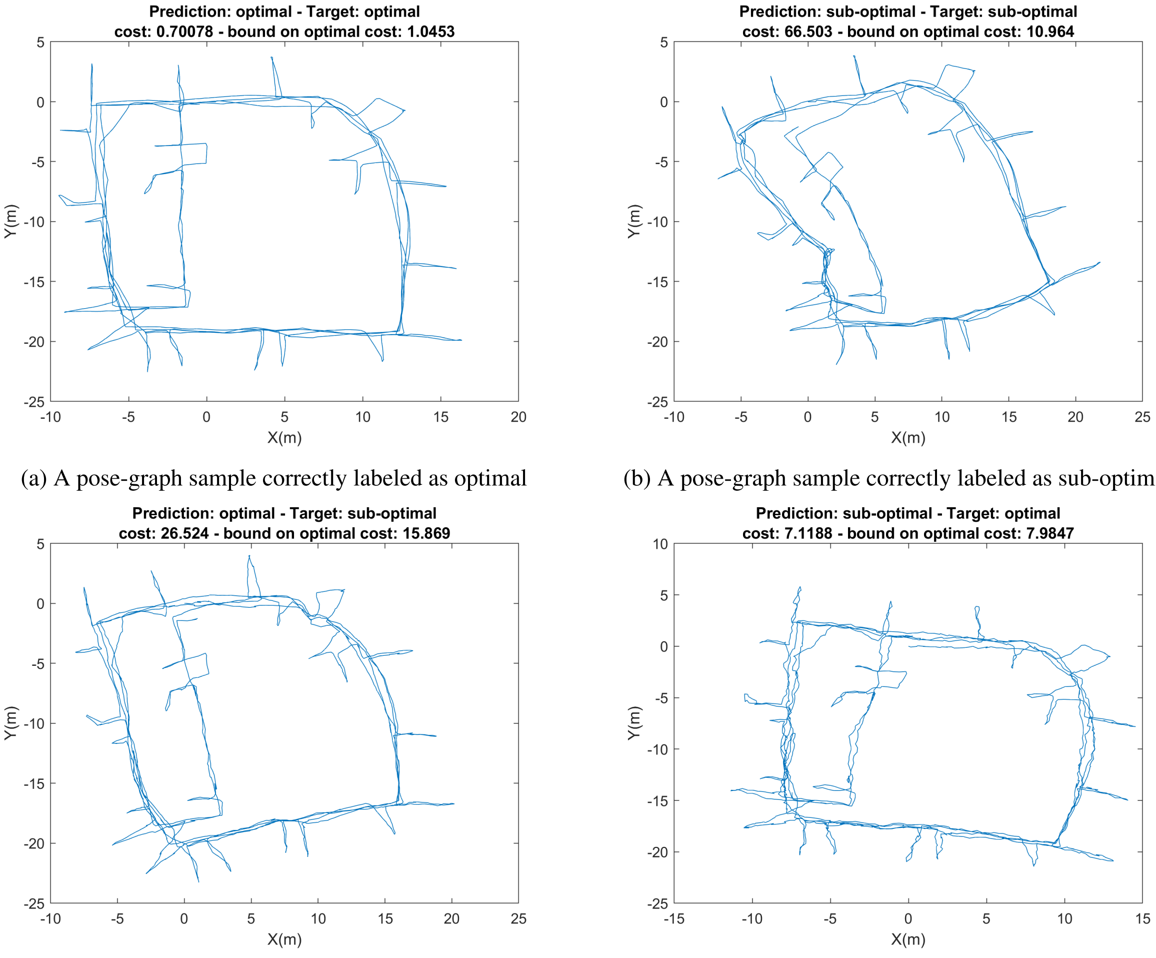


FIGURE 6. Leave-set-out cross-validation loss curves.

the training dataset. These variants exhibited different trajectory shapes and were obtained in different environments. This is attributed to the fact that the network is not aware of the raw x, y, and theta values, and hence is not aware of how the trajectory looks like. Rather, classification is done based on the cost feature of each node in the graph which reflects how much estimation error is associated with the edges linked to each node. Furthermore, GNNs have the ability to perform estimations on graphs whose sizes are different from those in the training set, as analyzed in [40] and confirmed through the various tests conducted in this paper.

Our proposed approach achieves a large speed-up when compared to other pose-graph optimality verification techniques such as solving for the SDP value [10], or computing lower and upper bounds on the optimal cost of the solution to the pose-graph as proposed in [13]. Figure 8 compares the time needed to compute the SDP values using SE-Sync [10] and to predict the optimality of a set of pose-graphs using our proposed approach. The number of nodes in the tested pose-graphs varies between 900 and 3500 nodes, while the number of edges ranges from 1100 to 5500 edges. For each pose-graph, 31 variants were generated where extra additive noise was added to orientation measurements. The standard deviation of the additive noise was varied from 0 to 0.3 in increments of 0.01. As can be noticed from the figure, the time needed to compute the SDP value increases when the noise and the size of the pose-graph (the sizes of the



(a) A pose-graph sample correctly labeled as optimal

(b) A pose-graph sample correctly labeled as sub-optimal

(c) A pose-graph sample incorrectly labeled as optimal

(d) A pose-graph sample incorrectly labeled as sub-optimal

FIGURE 7. Sample test results.

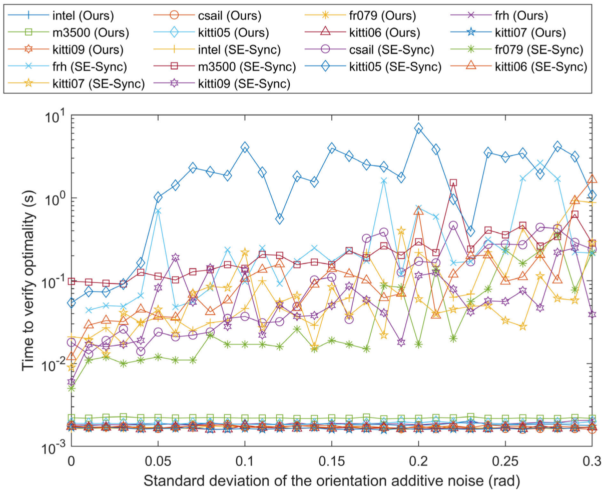


FIGURE 8. Time to verify optimality using our approach and SE-Sync.

pose-graphs are listed in Table 1) increase. Varying the noise parameters does not have any effect on the time needed to predict the optimality of a pose-graph using the proposed approach. Besides, increasing the size of the pose-graph introduces a very slight increase ($\sim 10^{-4}$ seconds between the smallest and largest tested pose-graphs) in the prediction time. In addition, our approach can be up-to one order of magnitude faster when performing predictions on batches of pose-graphs, which could be beneficial if a dedicated server is employed to perform optimality checks for multiple SLAM systems simultaneously. Table 2 lists the time needed to predict the optimality of batches of pose-graphs and the corresponding average time per sample.

TABLE 2. Time in seconds to predict optimality of batches of pose-graphs using our proposed approach.

Dataset	#Variants	Time for all variants (s)	Average time per variant (s)	#Edges
KITTI05	1130	0.46313	0.00041	2826
FRH	1028	0.38933	0.00038	2820
INTEL	842	0.24103	0.00029	1728
KITTI09	1130	0.21789	0.00019	1592
KITTI07	1164	0.20287	0.00017	1106
FR079	928	0.13150	0.00014	1217
CSAIL	990	0.13839	0.00014	1172
KITTI06	770	0.10361	0.00013	1150
Total	7982	1.88773		

While we do not have access to the source code of the approach proposed in [13], they reported the time needed to compute the lower and upper bounds on the optimal cost for some pose-graph datasets. These datasets include INTEL, FR079, CSAIL, M3500 [41], M3500a, M3500b, and M3500c[1] for which computing the lower and upper bounds consumed 2.2s, 2.3s, 1.5s, 20.9s, 23.4s, 25.1, and 24.2s respectively. As the size of the pose-graph and the magnitude of the additive noise increases, the time to compute the bounds also increases.

In conclusion, the time needed to classify a sample pose-graph as optimal or sub-optimal using the proposed approach could be shorter by up-to three orders of magnitude than SE-Sync and by up-to four orders of magnitude than the approach in [23], for the tested pose-graph variants. All the speed tests were conducted on an ASUS STRIX laptop, with Intel core i7-6700HQ @ 2.60GHz × 8. The C++ version of SE-Sync was used to measure the speed of computing the SDP value, while the proposed approach was implemented using the cpu version of the deep graph library [42]. Further speed-ups are expected using the gpu version of the library.

B. 10-FOLD CROSS-VALIDATION

The dataset was randomly split into 10 non-overlapping subsets, or folds, of equal size. The pose-graph neural classifier is then trained 10 times, where every fold serves as a test set exactly once, while the remaining folds are used as training sets. The overall performance of the model is then evaluated based on its resulting test accuracies. Figure 9 depicts the flow-chart of the 10-fold cross-validation technique.

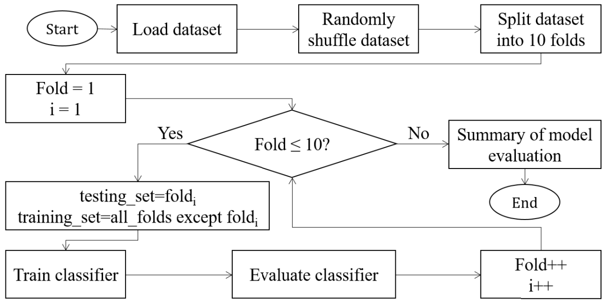


FIGURE 9. 10-fold cross-validation technique.

The loss curves obtained from training the model 10 times exhibited two different convergence speeds. The majority showed fast convergence while some demonstrated slower convergence, and hence need much longer training periods to converge. When training the model for a longer period of time, it was clear that the loss curve with slower convergence was still decreasing. The accuracies obtained on the test set, as well as on all the pose-graph variants' sets in all the training scenarios are shown in Figure 10. To improve the accuracies obtained for folds 3, 4, 6, and 10, the model should be trained for much longer.

To further prove the validity of the proposed model, the prediction accuracy of optimal and sub-optimal samples were computed. Table 3 lists the number and percentage of the correctly and incorrectly labeled samples from the test set during the 10-fold cross validation. The test set consisted of 818 samples that were different from one fold to another. It can be noticed that the prediction accuracy in the cases where the model was not yet converged is lower compared

[1] M3500a, M3500b, and M3500c are variants of the M3500 pose-graph where extra additive noise with standard deviation 0.1rad, 0.2rad, and 0.3rad was added to the relative orientation measurements.

TABLE 3. Prediction accuracy of optimal and sub-optimal samples obtained from the 10-fold cross validation test.

Fold	Correctly detected as optimal		Correctly detected as sub-optimal		Incorrectly Detected as optimal		Incorrectly detected as sub-optimal		Model converged?
	#samples	**%**	**#samples**	**%**	**#samples**	**%**	**#samples**	**%**	
1	388	47.43%	388	47.43%	25	3.06%	17	2.08%	Yes
2	336	41.08%	446	54.52%	15	1.83%	21	2.57%	Yes
3	337	41.20%	394	48.17%	55	6.72	32	3.91%	No
4	345	42.18%	392	47.92%	41	5.01%	40	4.89%	No
5	363	44.38%	414	50.61%	21	2.57%	20	2.44%	Yes
6	341	41.68%	371	45.35%	46	5.62%	60	7.33%	No
7	387	47.31%	394	48.17%	24	2.93%	13	1.59%	Yes
8	377	46.09%	405	49.51%	19	2.32%	17	2.08%	Yes
9	358	43.77%	420	51.34%	20	2.45%	20	2.45%	Yes
10	343	41.93%	367	44.87%	76	9.29%	32	3.91%	No

TABLE 4. Performance of pose-graph neural classifier with different numbers of PoseConv layers.

Set		**Number of PoseConv Layers**							
		1	**2**	**3**	**4**	**5**	**10**	**15**	**20**
Training Sets	**FRH**	94.20%	87.00%	93.70%	86.60%	94.30%	93.80%	94.00%	94.00%
	FR079	91.80%	82.00%	94.10%	86.40%	91.70%	92.60%	93.50%	94.30%
	INTEL	95.70%	88.00%	96.70%	88.80%	96.70%	96.40%	96.00%	95.70%
	KITTI07	97.40%	93.90%	97.00%	94.20%	97.60%	97.30%	97.70%	97.30%
	KITTI09	93.30%	88.90%	94.00%	87.70%	94.60%	94.90%	94.70%	94.50%
Testing Sets	**KITTI06**	**91.70%**	**83.20%**	**93.00%**	**85.10%**	**90.60%**	**92.50%**	**90.60%**	**92.30%**
	CSAIL	**94.30%**	**84.20%**	**93.80%**	**84.40%**	**92.70%**	**93.80%**	**93.30%**	**94.20%**

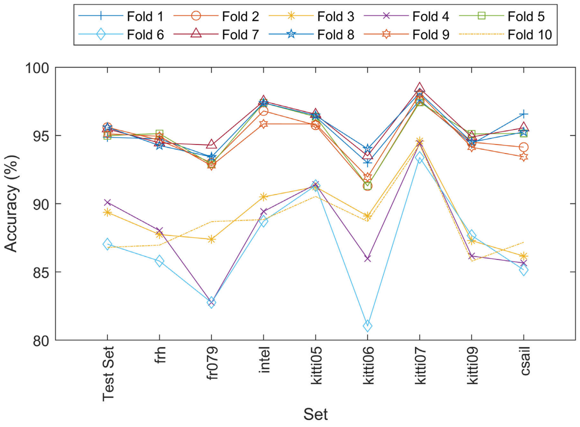


FIGURE 10. 10-fold cross-validation results - Accuracies.

to when the model has already converged. More particularly, the accuracies obtained for folds 3, 4, 6, and 10 are slightly less than the other folds. Allowing more training epochs for these folds will result in an increase in the prediction accuracy. All in all, the model was able to successfully classify both optimal and sub-optimal pose-graph candidate solutions with high accuracy.

C. NUMBER OF POSECONV LAYERS

In this section, the performance of the proposed neural network architecture will be compared to architectures with varying numbers of PoseConv layers. FRH, FR079, INTEL, KITTI07, and KITTI09 pose-graph variants were used as training sets, while KITTI06 and CSAIL pose-graph variants were used for testing. The number of PoseConv layers in the model were varied between 1 and 20 and each model was trained for 200 epochs. Samples of the resulting training and testing accuracies are listed in Table 4.

Across the 20 tests, it was noticed that some models converge much faster than others, as depicted in Figure 11. We believe that this happens because of the differences in the initial conditions. Figure 12 shows an example where the same model was trained twice with different initial random seeds for more than 8000 epochs. This figure suggests that both models are likely to eventually converge to similar loss values and hence achieve comparable performance.

The differences in the convergence speeds with varying initial random seeds could be attributed to the use of the chordal cost function to construct the messages exchanged among the nodes. In [43], it was shown that the chordal cost for a minimal 2D pose-graph problem with three poses and three measurements introduces some saddle points, which could slow down the training process. While the training process does not minimize chordal cost itself, and the result in [43] does not comment on the existence of saddle points in larger problems, we conjecture that the use of chordal cost in the messages contributes to the difference in convergence rate.

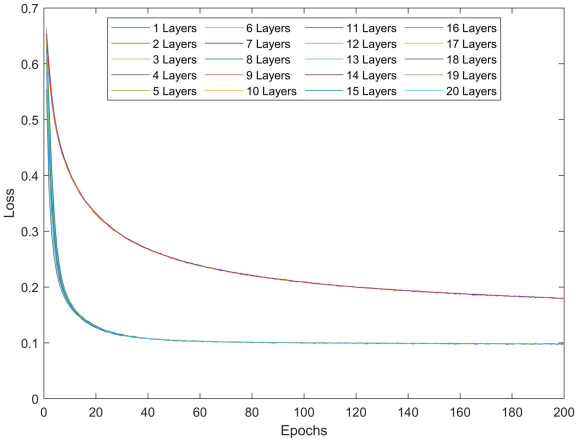


FIGURE 11. Loss curves for models with different number of PoseConv layers.

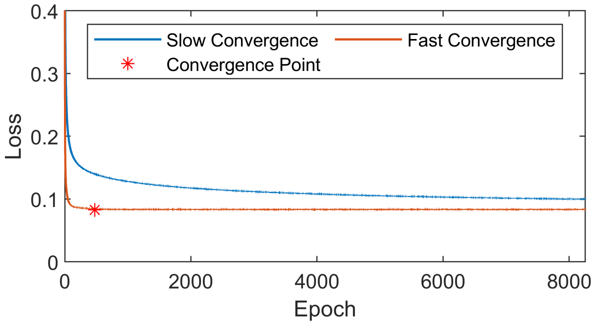


FIGURE 12. Loss curves obtained when training the same model with different initial seeds.

Moreover, the results achieved by the model with one PoseConv layer are comparable to those achieved by the models with more PoseConv layers. Hence, the proposed approach employs only one PoseConv layer to achieve higher efficiency and faster training and prediction.

V. CONCLUSION

In this paper, a novel pose-graph optimality verification technique was proposed. A graph convolutional-like layer, dubbed PoseConv, was developed and used in a graph neural network to address the problem of classifying pose-graphs into optimal or sub-optimal. PoseConv performs message passing between the nodes of a pose-graph and aggregates the exchanged messages in the *cost* feature of every node in the graph, while keeping the other node and edge features unchanged. The *cost* feature is a new feature that we proposed to add, to carry the information needed for the classifier to learn how to distinguish between optimal and sub-optimal solutions. Eight different bench-marking datasets were employed to build a dataset that includes pose-graphs of variable size, varied measurement noise parameters, and recorded in diverse environments using different equipment. The proposed approach has been extensively tested on various partitions of the dataset. Training and testing results have verified the validity and generalization of the proposed approach. The pose-graph neural classifier has been able to distinguish between optimal and sub-optimal solutions of unseen pose-graph samples from the dataset. This is due to the fact that the classifier classifies pose-graphs based on the cost features, which are independent of the actual values of the node and edge features of the pose-graph. This makes the approach capable of generalizing to previously unseen pose-graphs, even if they are of different sizes than those in the training dataset.

The work proposed in this paper can be extended to allow classification of candidate solutions to 3D pose-graphs into optimal and sub-optimal estimates. To accommodate this extension, the proposed PoseConv layer would be altered due to the increase in the dimensionality of the input pose-graph. Each node in the 3D pose-graph will have seven features, instead of four in the current version, that represent the 3D position and orientation of the robot and the additional *cost* feature that will accommodate the communicated messages. Consequently, the linking edges will encode the relative 3D pose between two nodes and the corresponding measurement noise parameters. Similar to the current implementation of the PoseConv layer, messages will carry information about the chordal cost (for the 3D case, instead) associated with each edge and will be aggregated into the *cost* feature of each node. The increase in the number of node and edge features might require a more complex architecture of the neural network.

Future research work in this area can also be directed towards integrating the proposed neural classifier with a correction mechanism where sub-optimal candidate solutions to pose-graph can be improved. These areas of improvements are left for future work.

REFERENCES

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics* (Intelligent Robotics and Autonomous Agents). Cambridge, MA, USA: MIT Press, 2005.

[2] R. Azzam, T. Taha, S. Huang, and Y. Zweiri, "Feature-based visual simultaneous localization and mapping: A survey," *Social Netw. Appl. Sci.*, vol. 2, no. 2, Feb. 2020, Art. no. 224, doi: 10.1007/s42452-020-2001-3.

[3] M. A. K. Niloy, A. Shama, R. K. Chakrabortty, M. J. Ryan, F. R. Badal, Z. Tasneem, M. H. Ahamed, S. I. Moyeen, S. K. Das, M. F. Ali, M. R. Islam, and D. K. Saha, "Critical design and control issues of indoor autonomous mobile robots: A review," *IEEE Access*, vol. 9, pp. 35338–35370, 2021.

[4] R. Azzam, Y. Alkendi, T. Taha, S. Huang, and Y. Zweiri, "A stacked LSTM-based approach for reducing semantic pose estimation error," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–14, 2021.

[5] F. Bai, T. Vidal-Calleja, and G. Grisetti, "Sparse pose graph optimization in cycle space," *IEEE Trans. Robot.*, early access, Feb. 3, 2021, doi: 10.1109/TRO.2021.3050328.

[6] M. Mazuran, G. D. Tipaldi, L. Spinello, W. Burgard, and C. Stachniss, "A statistical measure for map consistency in SLAM," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 3650–3655.

[7] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.

[8] S. Huang, Y. Lai, U. Frese, and G. Dissanayake, "How far is SLAM from a linear least squares problem?" in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 3011–3016.

[9] S. Huang and G. Dissanayake, "A critique of current developments in simultaneous localization and mapping," *Int. J. Adv. Robot. Syst.*, vol. 13, no. 5, Sep. 2016, Art. no. 172988141666948, doi: 10.1177/1729881416669482.

[10] D. Rosen, L. Carlone, A. Bandeira, and J. Leonard, "SE-Sync: A certifiably correct algorithm for synchronization over the special Euclidean group," Comput. Sci. Artif. Intell. Lab., Massachusetts Inst. Technol., Cambridge, MA, USA, Tech. Rep. MIT-CSAIL-TR-2017-002, Feb. 2017.

[11] J. Briales and J. Gonzalez-Jimenez, "Fast global optimality verification in 3D SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 4630–4636.

[12] J. Briales and J. Gonzalez-Jimenez, "Cartan-sync: Fast and global SE(d)-synchronization," *IEEE Robot. Autom. Lett.*, vol. 2, no. 4, pp. 2127–2134, Oct. 2017.

[13] L. Carlone and F. Dellaert, "Duality-based verification techniques for 2D SLAM," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 4589–4596.

[14] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[15] W. Cao, Z. Yan, Z. He, and Z. He, "A comprehensive survey on geometric deep learning," *IEEE Access*, vol. 8, pp. 35929–35949, 2020.

[16] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *CoRR*, vol. abs/1812.08434, pp. 1–25, Dec. 2018. [Online]. Available: http://arxiv.org/abs/1812.08434

[17] S. Huang, H. Wang, U. Frese, and G. Dissanayake, "On the number of local minima to the point feature based SLAM problem," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 2074–2079.

[18] L. Carlone, "A convergence analysis for pose graph optimization via Gauss–Newton methods," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 965–972.

[19] L. Carlone, D. M. Rosen, G. Calafiore, J. J. Leonard, and F. Dellaert, "Lagrangian duality in 3D SLAM: Verification techniques and optimal solutions," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 125–132.

[20] L. Carlone, G. C. Calafiore, C. Tommolillo, and F. Dellaert, "Planar pose graph optimization: Duality, optimal solutions, and verification," *IEEE Trans. Robot.*, vol. 32, no. 3, pp. 545–565, Jun. 2016.

[21] A. Eriksson, C. Olsson, F. Kahl, and T.-J. Chin, "Rotation averaging with the chordal distance: Global minimizers and strong duality," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 1, pp. 256–268, Jan. 2021.

[22] R. Hartley, J. Trumpf, Y. Dai, and H. Li, "Rotation averaging," *Int. J. Comput. Vis.*, vol. 103, no. 3, pp. 267–305, Jul. 2013.

[23] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge Univ. Press, 2004.

[24] D. M. Rosen, L. Carlone, A. S. Bandeira, and J. J. Leonard, "SE-sync: A certifiably correct algorithm for synchronization over the special Euclidean group," *Int. J. Robot. Res.*, vol. 38, nos. 2–3, pp. 95–125, Mar. 2019.

[25] Y. Tian, K. Khosoussi, D. M. Rosen, and J. P. How, "Distributed certifiably correct pose-graph optimization," *IEEE Trans. Robot.*, early access, May 1, 2021, doi: 10.1109/TRO.2021.3072346.

[26] P. W. Battaglia *et al.*, "Relational inductive biases, deep learning, and graph networks," *CoRR*, vol. abs/1806.01261, pp. 1–40, Jun. 2018. [Online]. Available: http://arxiv.org/abs/1806.01261

[27] T. Zhong, T. Wang, J. Wang, J. Wu, and F. Zhou, "Multiple-aspect attentional graph neural networks for online social network user localization," *IEEE Access*, vol. 8, pp. 95223–95234, 2020.

[28] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei, "Relation networks for object detection," *CoRR*, vol. abs/1711.11575, pp. 1–11, Nov. 2017. [Online]. Available: http://arxiv.org/abs/1711.11575

[29] D. Lin, J. Lin, L. Zhao, Z. J. Wang, and Z. Chen, "Multilabel aerial image classification with a concept attention graph neural network," *IEEE Trans. Geosci. Remote Sens.*, early access, Mar. 2, 2021, doi: 10.1109/TGRS.2020.3041461.

[30] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 7794–7803.

[31] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Represent. (ICLR)*, Toulon, France, Apr. 2017, pp. 1–14. [Online]. Available: https://openreview.net/forum?id=SJU4ayYgl

[32] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *CoRR*, vol. abs/1801.07829, pp. 1–13, Jan. 2018. [Online]. Available: http://arxiv.org/abs/1801.07829

[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, pp. 1–15, Dec. 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[34] R. Kummerle. (2009). *Slam Benchmarking Webpage*. [Online]. Available: http://ais.informatik.uni-freiburg.de/slamevaluation and https://ci.nii.ac.jp/naid/10030542269/en/

[35] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013, doi: 10.1177/0278364913491297.

[36] D. Rosen and L. Carlone, "Computational enhancements for certifiably correct SLAM," presented at the Int. Conf. Intell. Robots Syst. Workshop Introspective Methods Reliable Autonomy (IROS), Sep. 2017.

[37] D. M. Rosen, L. Carlone, A. S. Bandeira, and J. J. Leonard, "A certifiably correct algorithm for synchronization over the special Euclidean group," in *Proc. Int. Workshop Algorithmic Found. Robot. (WAFR)*, San Francisco, CA, USA, Dec. 2016, pp. 64–79.

[38] P.-A. Absil, C. G. Baker, and K. A. Gallivan, "Trust-region methods on Riemannian manifolds," *Found. Comput. Math.*, vol. 7, no. 3, pp. 303–330, Jul. 2007.

[39] R. Caruana, S. Lawrence, and C. Giles, "Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping," in *Advances in Neural Information Processing Systems*, vol. 13, T. Leen, T. Dietterich, and V. Tresp, Eds. Cambridge, MA, USA: MIT Press, 2001. [Online]. Available: https://proceedings.neurips.cc/paper/2000/file/059fdcd96baeb75112f09fa1dcc740cc-Paper.pdf

[40] L. Ruiz, F. Gama, and A. Ribeiro, "Graph neural networks: Architectures, stability, and transferability," *Proc. IEEE*, vol. 109, no. 5, pp. 660–682, May 2021.

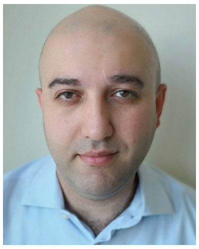
[41] E. Olson, J. Leonard, and S. Teller, "Fast iterative alignment of pose graphs with poor initial estimates," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2006, pp. 2262–2269.

[42] M. Wang, L. Yu, D. Zheng, Q. Gan, Y. Gai, Z. Ye, M. Li, J. Zhou, Q. Huang, C. Ma, Z. Huang, Q. Guo, H. Zhang, H. Lin, J. Zhao, J. Li, A. J. Smola, and Z. Zhang, "Deep graph library: Towards efficient and scalable deep learning on graphs," in *Proc. ICLR Workshop Represent. Learn. Graphs Manifolds*, 2019, pp. 1–7. [Online]. Available: https://arxiv.org/abs/1909.01315

[43] F. H. Kong, J. Zhao, L. Zhao, and S. Huang, "Analysis of minima for geodesic and chordal cost for a minimal 2-D pose-graph SLAM problem," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 323–330, Apr. 2020.

RANA AZZAM received the B.Sc. degree in computer engineering and the M.Sc. degree by Research in electrical and computer engineering from Khalifa University of Science and Technology, Abu Dhabi, United Arab Emirates, in 2014 and 2016, respectively, and the Ph.D. degree in engineering with the KU Center for Autonomous Robotic Systems (KUCARS), Khalifa University of Science and Technology, in 2020, with a focus on robotics. She is currently a Postdoctoral Research Fellow with the Department of Aerospace Engineering, Khalifa University of Science and Technology. Her current research interests include artificial intelligence and deep learning, reinforcement learning, and mobile robots simultaneous localization and mapping (SLAM).

FELIX H. KONG received the Ph.D. degree from the Australian Centre for Field Robotics (ACFR), University of Sydney (UTS), in 2019. He is currently a Postdoctoral Fellow with the Centre for Autonomous Systems, UTS. His interests include optimization-based robotics applications, including robot localization and mapping, iterative learning control, and ocean flow field estimation.



TAREK TAHA received the M.Eng. degree in computer control and the Ph.D. degree from the Centre of Excellence for Autonomous Systems (CAS), University of Technology Sydney, Australia, in 2004 and 2012, respectively. From 2008 to 2013, he worked as a Senior Mechatronics Engineer with a Sydney based Engineering Research and Development Company, before joining Khalifa University, from 2014 to 2018. He then led Algorythma's Autonomous Aerial Lab, before joining Dubai Future Foundation to lead the Robotics Laboratory. His research interests include autonomous exploration, navigation and mapping, machine vision, human–robot interaction, assistive robotics, and artificial intelligence.

YAHYA ZWEIRI received the Ph.D. degree from King's College London, in 2003. He was involved in defense and security research projects in the last 20 years with the Defence Science and Technology Laboratory, King's College London, and King Abdullah II Design and Development Bureau, Jordan. He is currently the School Director of the research and enterprise with Kingston University, London, U.K. He is also an Associate Professor with the Department of Aerospace, Khalifa University of Science and Technology, United Arab Emirates. He has published over 100 refereed journal and conference articles and filed six patents, in USA and U.K., in unmanned systems field. His research interests include interaction dynamics between unmanned systems and unknown environments by means of deep learning, machine intelligence, constrained optimization, and advanced control.