Learning Embeddings for Loop Closing Detection Using Graph Neural Network

Abhishek Khoyani[1] and Marzieh Amini[2]

***Abstract*— Loop closure detection involves matching a current query image with images from an existing database during simultaneous localization and mapping (SLAM). In graph-based SLAM, each pose is defined by graph nodes, and relative transformations are established through connecting edges. Although various methods employ graph-based algorithms for loop closure, they differ from the representation in graph SLAM. In most cases, features from query and reference images are utilized to compute similarity. This paper proposes the supervised method by exploiting the graph structure with graph neural network (GNN) to learn a new embeddings on top of intermediate features extracted from convolutional models that can enhance the similarity score for the loop closure queries and matching nodes whereas diminishing the score for dissimilar nodes, finally resulting in reduced false positives and false negatives. GNN makes it possible to learn a new features by gathering neighborhood information that helps to effectively distinguishing between similarity and dissimilarity. The proposed embeddings are evaluated in comparison with other convolution-based features, revealing their efficacy in enhancing evaluation metrics and overall performance.**

I. INTRODUCTION

Loop closing detection (LCD) or visual place recognition (VPR) is a crucial part of the simultaneous localization and mapping (SLAM) algorithm that matches non-sequential entries in the estimated graph. This process helps to prevent the generation of false and redundant graphs and reduces the overall drift error. The graph structure is a powerful non-linear data structure that can effectively represent various real-world problems, including bio-chemical networks, social media networks, traffic signal networks, and maps, among others. By leveraging state-of-the-art GNN [1] techniques, we can learn the relationships between nodes and their neighbors to make predictions about node attributes (node classification), node linkages (edge prediction), or even overall graph properties (graph classification). In the context of loop closing, the goal is to predict the similarity between a query node and a matching node. While many existing approaches focus solely on the features extracted from the query and matching nodes, in [2], a conventional graph diffusion mechanism was proposed to learn features from the surrounding neighbors. In our previous work [3], we have constructed a graph from the dataset and computed the cosine similarity directly from the features extracted using a convolutional backbone. This paper presents a novel approach to learn embeddings, utilizing a graphSAGE [4] layer, which is a variant of graph convolutional neural networks. The contributions of this work can be summarized as follows:

- Investigating the applicability of GNN in the domain of loop closing detection, potentially pioneering this approach in the literature.
- Formulating the loop closing problem as a link prediction problem, allowing us to learn similarity and dissimilarity embeddings from prior neighbourhood data.

This work was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada.

[1]Abhishek Khoyani is with Department of System and Computer Engineering, Carleton University, Ottawa, ON, Canada.abhishekkhoyani@cmail.carleton.ca

[2]Marzieh Amini is with the School of Information Technology, Carleton University, Ottawa, ON, Canada.marzieh.amini@carleton.ca

II. RELATED WORKS

A number of graph-based algorithms have been proposed in the literature, but they do not construct or replicate a graph structure that can be used in graph SLAM. For instance in [5], a proximity graph was proposed for visual vocabulary called hierarchical navigable small world (HNSW) instead of BoW. Graph diffusion was implemented in [2] to pass feature information to the neighboring nodes and checked temporal consistency to avoid false positives. A graph-based image representation was proposed in X-view [6], which incorporated both the geometry and semantics of the scene. Same kind of approach was followed in SymbioLCD2 [7]. To learn the spatial relationship between extracted features from the image, SymbioLCD2 generated multi-tier graph from the features and compared them to predict the loop closure.

III. METHODOLOGY

Link prediction is a supervised binary classification problem that involves predicting whether there should be a connection between two nodes within a graph [8]. In the context of loop closing detection, the objective is to predict the similarity between two images and subsequently add an edge to the graph based on this similarity [9]. In order to train a GNN in a supervised manner, the edges in the graph are divided into separate train, validation, and test datasets. The validation and test datasets exclusively contain "supervision" edges, which are utilized for evaluation purposes. On the other hand, the train dataset consists of both "supervision" edges and "message-passing" edges. The "message-passing" edges are utilized during training to facilitate the learning of neighborhood information.

A. Dataset

The dataset preparation involved using five sequences of image-to-image data along with their corresponding ground truth to construct a graph. Table I provides the summary of

TABLE I: Datasets Summary

Dataset	Description	Image Resolution	# Images	# Loop Closure Matches
New College (NC - reduced) [10]	Outdoor, dynamic	480×640	394	640
City Center (CC - reduced) [10]	Urban, dynamic	480×640	274	417
KAIST North (KN) [11]	Outdoor, Day & Night	900×1200	204	510
KAIST East (KE) [11]	Outdoor, Day & Night	900×1200	200	500
KAIST West (KW) [11]	Outdoor, Day & Night	900×1200	200	500

the five sequence and respective groundtruth. Groundtruth for each sequence is received from [12]. By combining these five datasets, a graph with a total of 1272 nodes and 2567 positive edges was created. To simplify the process, the graph was constructed using features extracted from ShuffleNet, rather than the original images. It is important to note that for this problem, there are a total of 1,616,712 possible edges between the nodes (1272×1271), out of which only 2567 are positive edges. This characteristic makes the problem highly unbalanced in terms of classification. To address this issue, negative edges are dynamically added during training based on the ratio of total possible edges to positive edges. The dataset was split into train, validation, and test sets in a proportion of 80%, 10%, and 10%, respectively. Considering the imbalanced nature of the problem, evaluation metrics such as specificity, precision, and F1-score were used. Specificity helps to assess the model's ability to identify dissimilarities between images, precision measures the model's ability to correctly predict true loop closures with minimal false positives, and the overall evaluation is determined based on the F1-score, which takes into account the data imbalance.

IV. EXPERIMENTAL RESULTS

The primary objective of utilizing a GNN is to learn new embeddings on top of convolutional backbone to effectively distinguish between similar and dissimilar nodes. This distinction is crucial as it allows the embeddings' cosine similarity scores to accurately reflect the level of similarity between nodes, leading to improved evaluation metrics. The basic architecture, as depicted in Fig. 1, involves adding two graphSAGE [4] layers with intermediate ReLU activation and dropout layers on top of the ShuffleNet features. GraphSAGE layers accumulates the features from the neighbor nodes from the training set which exhibits the higher similarity and learns a new embeddings during training phase to differentiate similar and dissimilar neighbors. The output of the final graph convolutional layer represents the learned embeddings, which are subsequently used to predict loop closures by calculating the cosine similarity between pairs of vertices. The number of nodes in each graph convolutional layer and the dropout coefficient are determined through a hyperparameter tuning strategy [13], aiming to optimize the model's performance.

The selection of hyperparameters is performed using the wandb sweep agent, which iteratively runs the training loop with different combinations of input parameter settings. The goal is to optimize a specific metric, in this case, the cosine embedding loss [14] on the validation set. The optimization process spans 40 epochs. Various hyperparameters, including the number of nodes in each graph convolutional layer and the dropout coefficient for the model architecture, are tuned. Additionally, hyperparameters related to the training strategy, such as the initial learning rate, L2 normalization (decay factor in the Adam optimizer) [15], and the patience number to reduce the learning rate on a plateau, are also optimized. After conducting multiple runs, the values presented in Table II are finalized as the hyperparameter settings for the final experiments.

TABLE II: Final hyperparameter values selected after 128 iteration monitored to minimize validation loss.

Hyperparameter	**Value**
GCN-1	1000
GCN-2	2000
Learning Rate	1e-3
Dropout	0.5
Decay	1e-3

Once the final model architecture parameters and initial training parameters are established, the model is trained till the validation loss is reduced to 0.08364.

The final prediction results on the test set demonstrate a specificity of 96.48%, precision of 95.81%, and F1-score of 87.47%. These results indicate that the proposed model effectively reduces false positives while improving true positives and true negatives.

Fig. 2 presents a comparison between the results obtained from the direct extracted features from the ShuffleNet and the results achieved using the proposed GNN method on the dataset. Notably, Fig. 2 reveals that the proposed GNN models outperform the ConvNet-based approach, achieving an average precision of 93.94% which is at least 20% higher than the average precision of 73.85% achieved by ShuffleNet extracted features.

It is important to note that the proposed models require the availability of some neighboring information initially in order to generate suitable embeddings. As a result, these models are not suitable for real-time applications. However, they can be employed offline to enhance the results obtained using online methods, particularly by reducing the occurrence of falsely identified loop closures, which can have detrimental effects on the overall SLAM system.

V. CONCLUSIONS

In conclusion, this paper has focused on the implementation and evaluation of a GNN approach for loop closing detection in SLAM. The challenge of predicting loop closures in visual SLAM systems has been successfully

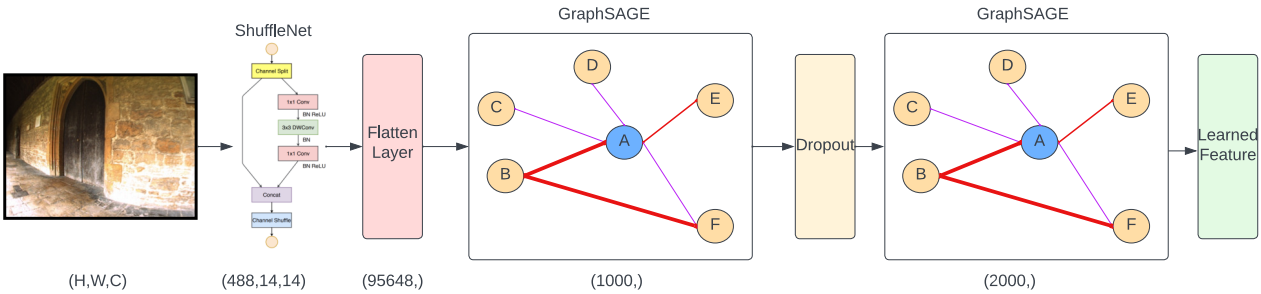

Fig. 1: Model architecture to learn the embeddings using graphSAGE layer on top of ShuffleNet.

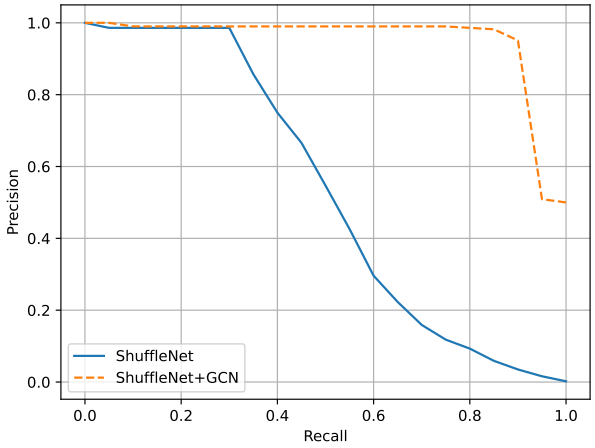

Fig. 2: The performance comparison of GNN versus vanilla backbone models in terms of PR curves.

addressed by leveraging the power of graph structures and GNN techniques. Through the use of learned embeddings and cosine similarity, our proposed method demonstrates improved performance in separating similar and dissimilar nodes, leading to enhanced evaluation metrics, specifically precision and specificity.

References

[1] B. Sanchez-Lengeling, E. Reif, A. Pearce, and A. B. Wiltschko, "A gentle introduction to graph neural networks," *Distill*, 2021, https://distill.pub/2021/gnn-intro.

[2] X. Zhang, L. Wang, Y. Zhao, and Y. Su, "Graph-based place recognition in image sequences with cnn features," *Journal of Intelligent & Robotic Systems*, vol. 95, pp. 389–403, 2019.

[3] A. Khoyani and M. Amini, "Real-time loop closure detection with constructing graph," *submitted to IEEE Sensors Journal*, 2023.

[4] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.

[5] S. An, G. Che, F. Zhou, X. Liu, X. Ma, and Y. Chen, "Fast and incremental loop closure detection using proximity graphs," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 378–385.

[6] A. Gawel, C. Del Don, R. Siegwart, J. Nieto, and C. Cadena, "X-view: Graph-based semantic multi-view localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1687–1694, 2018.

[7] J. J. Kim, M. Urschler, P. J. Riddle, and J. S. Wicker, "Closing the loop: Graph networks to unify semantic objects and visual features for multi-object scenes," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 4352–4358.

[8] D. Liben-Nowell and J. Kleinberg, "The link prediction problem for social networks," in *Proceedings of the twelfth international conference on Information and knowledge management*, 2003, pp. 556–559.

[9] K. A. Tsintotas, L. Bampis, and A. Gasteratos, "The revisiting problem in simultaneous localization and mapping: A survey on visual loop closure detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 11, pp. 19 929–19 953, 2022.

[10] M. Cummins and P. Newman, "Fab-map: Probabilistic localization and mapping in the space of appearance," *International Journal of Robotics Research*, vol. 27, pp. 647–665, 6 2008.

[11] Y. K. Choi, N. I. Kim, K. B. Park, S. M. Hwang, J. S. Yoon, and I. S. Kweon, "All-day visual place recognition: Benchmark dataset and baselines," in *CVPR2015 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society and the Computer Vision Foundation (CVF), 2015.

[12] D.-W. Shin, Y.-S. Ho, and E.-S. Kim, "Loop closure detection in simultaneous localization and mapping using descriptor from generative adversarial network," *Journal of Electronic Imaging*, vol. 28, no. 1, pp. 013 014–013 014, 2019.

[13] S. Falkner, A. Klein, and F. Hutter, "Bohb: Robust and efficient hyperparameter optimization at scale," in *International conference on machine learning*. PMLR, 2018, pp. 1437–1446.

[14] B. Barz and J. Denzler, "Deep learning on small datasets without pre-training using cosine loss," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2020, pp. 1371–1380.

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.