Deep Sets Are Viable Graph Learners

Gerrit Großmann[1]

Saarland Informatics Campus, DFKI, Saarbrücken, Germany
gerrit.grossmann@dfki.de

1 Introduction

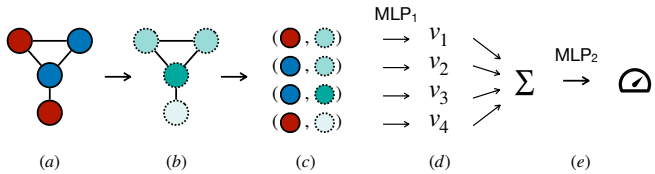

Fig. 1. Method Overview. (a): A 4-node input graph featuring binary node attributes (red, blue). (b): Positional embeddings generated from the adjacency matrix are visualized as varying shades of teal. (c): Concatenated with the original node attributes yields the multiset graph abstraction. (d): A first MLP transforms each node vector into a latent representation. (e): A summation of these latent representations is fed into a second MLP to produce the final prediction.

DeepSet neural networks (here: DSNNs) perform prediction tasks on (multi)sets: given a sequence of data points, their prediction is guaranteed to be invariant to their concrete ordering [14]. This study explores the application of DSNNs for predicting graph-based properties, a role typically filled by Graph Neural Networks (GNNs) using message-passing architectures. To achieve this, we employ graph centrality measures as positional encodings and evaluate their expressiveness. We find that *DeepSet*-based graph predictions are comparable in performance to established GNN models, while also ensuring invariance to node order. Our objective is not to introduce a new architecture for surpassing SOTA results. Instead, we aim to offer a fresh perspective on the significance and function of message-passing techniques, while revealing the extent to which simple metrics can encapsulate complex graph topologies.

DeepSets. Assume a multiset X on which we want to perform a prediction. A first *multi-layer perceptron* (MLP) maps each element $x \in X$ to a latent representation ($\mathrm{MLP}_1 : X \to \mathbb{R}^h$, where h is the latent dimension). A second MLP maps the sum of all latent representations to a final output ($\mathrm{MLP}_2 : \mathbb{R}^h \to \mathbb{R}^o$, assuming o is the output dimension). Together this gives raise to the following equation [12] of a DSNN:

$$y = \mathrm{MLP}_2\Big(\sum_{x \in X} \mathrm{MLP}_1(x)\Big)\,. \tag{1}$$

DSNNs are universal approximators and guaranteed to yield the same output regardless of the concrete ordering in which the elements of X are processed [14].

Method Overview. We lift DSNNs to graphs using positional encodings. Specifically, we compute a permutation-invariant positional encoding for each node. Next, we concatenate node feature vectors and positional encoding. Finally, we use DSNNs to perform a prediction on the multiset of nodes (cf. Figure 1).

Related Work. State-of-the-art methods in graph machine learning primarily employ GNNs, that transmit messages between nodes [3]. These messages are either exchanged along the edges of the input graph or, in newer Graph Transformers, between all nodes [13]. Graph Transformers have also sparked research into positional encodings on graphs [7,1,6,2,15]. Transformers can be considered a specialized implementation of *DeepSets*, albeit with a design distinct from DSNNs. This study bears similarity to work of [5], which employs a standard transformer on both nodes and edges (while we focus only on nodes). Even more pertinent is the research by [8], which also utilizes *DeepSets* but instead of using *permutation-equivariant* encodings, they use eigenvectors of the graph Laplacian, thereby undermining permutation-invariance of the final prediction.

Code Availability. Code is made available at `github.com/GerritGr/DSNN`.

2 Method

We assume a supervised learning task involving a set of graphs, each annotated with a ground-truth label. Each graph consists of n nodes and is represented by a symmetric, binary adjacency matrix $\mathbf{A} \in \{0,1\}^{n \times n}$ and a node-feature matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$. Edge features are disregarded for simplicity. We compute positional embeddings $\mathbf{P} \in \mathbb{R}^{n \times l}$ based solely on $\mathbf{A}$. Therefore, we use centrality metrics from the `NetworkX` library [4] to calculate the shortest path length from each node to the most central one We also use the sum of neighboring positional embeddings. Row-wise concatenation of $\mathbf{X}$ and $\mathbf{P}$ results in the lifted node-features $\mathbf{X}' \in \mathbb{R}^{n \times (d+l)}$. Finally, we deploy a DSNN to predict the graph labels using $\mathbf{X}'$, treating each row as an element of a multiset.

2.1 Expressiveness

The expressiveness of GNNs is typically measured by their ability to distinguish (i.e., make different predictions on) non-isomorphic graphs (isomorphic graphs will always lead to the same prediction by design). A well-known finding asserts that traditional GNNs are no more powerful than the first-order *Weisfeiler-Lehman* (1-WL) test [11,10]. Our study, complemented by Figure 2, reveals that:

1. There exist non-isomorphic graphs that are distinguishable by DSNNs but not by standard 1-WL GNNs. (This even holds for the more powerful 3-WL test, e.g., using the *Shrikhande* and *Rook's* 4×4 graph [15].)
2. There exist non-isomorphic graphs that are distinguishable by 1-WL GNNs but not by DSNNs.

Additionally, we observe that incorporating node features $\mathbf{X}$ into positional encodings enhances their expressive power (e.g., by introducing edge weights). Of course, one can also boost the expressiveness of GNNs by appending these positional encodings to

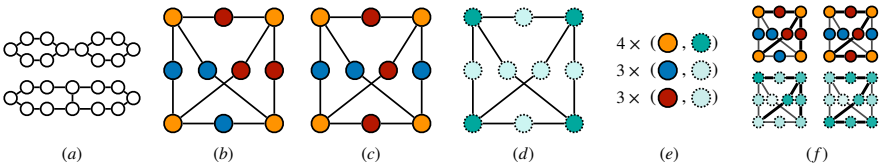

Fig. 2. Exploring Expressiveness. (a): Two graphs without node features indistinguishable by 1-WL and classical GNNs [10]. However, these graphs are distinguishable by DSNNs due to variances in betweenness centrality scores. (b) and (c) depict graphs with three types of nodes (red, blue, yellow) that are indistinguishable when analyzed by DSNNs but can be differentiated by 1-WL methods. (d): This occurs because the positional encodings produce two equivalence classes of automorphic nodes and the distribution of features within each classe, shown in (e), is identical. (f): When positional encodings are adapted to consider node features (by weighting different types of edges) the graphs again become distinguishable via DSNNs.

node features prior to message passing. Our findings somewhat contradict the claims in [8], demonstrating that relying solely on $\mathbf{A}$ is insufficient for generating positional encodings that enable universal approximations.

Open Problems. We identify three open problems for further exploration: (1) Is there an inherent limitation to the expressiveness of DSNNs when employing (permutation-equivariant) centrality measures? (Notably, betweenness centrality has been shown to be more powerful than PageRank.) (2) Are edge-weighted DSNNs strictly more powerful than 1-WL GNNs? (3) At which level l is the l-WL-test strictly more powerful than DSNNs (standard or edge-weighted)?

3 Results

We compare our *DeepSet*-based predictions (DSNN) with those generated by GCN and GIN implementations. We also report SOTA results from `paperswithcode.com`. As expected, the SOTA accuracy is significantly higher, due to specialized architecture tuning for the specific problem and leveraging of additional data (e.g., 3D coordinates, pre-training). In our experiments, we maintain a consistent architecture and set of hyperparameters across all tasks. Our method demonstrates performance comparable to well-established GNNs on the most prevalent graph learning tasks, as evidenced by the TUdataset [9]. Additional details can be found in the repository.

Table 1. Results: [Higher is better] Accuracy of different graph machine learning methods.

Dataset	DSNN	GCN	GIN	PNA	SOTA	⌀ # nodes	⌀ # edges	# graphs
PROTEIN	0.8	0.62	0.64	0.65	0.85	39.1	145.6	1113
MUTAG	1.0	0.79	0.89	0.53	1.0	17.9	39.6	188
ENZYMES	0.54	0.35	0.41	0.5	0.78	32.6	124.2	600
IMBD-binary	0.78	0.44	0.44	0.56	0.96	19.8	193.1	1000

4 Conclusions and Future Work

We conclude that the DSNN's ability to rival well-established methods without additional tuning is noteworthy, particularly given its conceptual simplicity and computational efficiency. This underscores the need for ongoing research to further understand their effectiveness in capturing the intrinsic topology of data. Specifically, exploring the strengths and limitations of centrality measures as well as alternative positional embeddings emerges as a valuable research direction.

References

1. Dwivedi, V.P., Luu, A.T., Laurent, T., Bengio, Y., Bresson, X.: Graph Neural Networks with Learnable Structural and Positional Representations (Feb 2022), arXiv:2110.07875 [cs]
2. Geisler, S., Li, Y., Mankowitz, D.J., Cemgil, A.T., Günnemann, S., Paduraru, C.: Transformers meet directed graphs. In: International Conference on Machine Learning. pp. 11144–11172. PMLR (2023)
3. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: International conference on machine learning. pp. 1263–1272. PMLR (2017)
4. Hagberg, A., Swart, P., S Chult, D.: Exploring network structure, dynamics, and function using networkx. Tech. rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States) (2008)
5. Kim, J., Nguyen, D., Min, S., Cho, S., Lee, M., Lee, H., Hong, S.: Pure transformers are powerful graph learners. Advances in Neural Information Processing Systems 35, 14582–14595 (2022)
6. Kreuzer, D., Beaini, D., Hamilton, W.L., Létourneau, V., Tossou, P.: Rethinking Graph Transformers with Spectral Attention. arXiv (2021)
7. Lim, D., Robinson, J., Zhao, L., Smidt, T., Sra, S., Maron, H., Jegelka, S.: Sign and Basis Invariant Networks for Spectral Graph Representation Learning (Sep 2022), arXiv:2202.13013 [cs, stat]
8. Ma, G., Wang, Y., Wang, Y.: Universal graph neural networks without message passing. (We also refer to the discussion on OpenReview.) (2022), `https://openreview.net/forum?id=P0bfBJaD4KP`
9. Morris, C., Kriege, N.M., Bause, F., Kersting, K., Mutzel, P., Neumann, M.: Tudataset: A collection of benchmark datasets for learning with graphs. arXiv preprint arXiv:2007.08663 (2020)
10. Morris, C., Ritzert, M., Fey, M., Hamilton, W.L., Lenssen, J.E., Rattan, G., Grohe, M.: Weisfeiler and leman go neural: Higher-order graph neural networks. In: Proceedings of the AAAI conference on artificial intelligence. vol. 33, pp. 4602–4609 (2019)
11. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018)
12. Xu, K., Li, J., Zhang, M., Du, S.S., Kawarabayashi, K.i., Jegelka, S.: What can neural networks reason about? arXiv preprint arXiv:1905.13211 (2019)
13. Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., Liu, T.Y.: Do transformers really perform badly for graph representation? Advances in Neural Information Processing Systems 34, 28877–28888 (2021)
14. Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R.R., Smola, A.J.: Deep sets. Advances in neural information processing systems 30 (2017)
15. Zhu, W., Wen, T., Song, G., Wang, L., Zheng, B.: On structural expressive power of graph transformers. arXiv preprint arXiv:2305.13987 (2023)