Pretraining Neural Architecture Search Controllers with Locality-based Self-Supervised Learning

Kwanghee Choi[1*] kwanghee.choi@hpcnt.com Minyoung Choe[2*] min05@sogang.ac.kr Hyelee Lee[3*] hyelee.lee@navercorp.com *Equal contributions.

[1]Hyperconnect [2]Sogang University [3]Naver

Introduction

- NAS (Neural Architecture Search) task is computationally expensive since obtaining the training set (architecture-performance pairs) is intrinsically expensive.
- We introduce a cheap self-supervised pretraining task that learns the architecture embedding using graph edit-distance.
- We apply our scheme to NAO (Neural Architecture Optimization) and show that pretraining boosts the performance.

Our method

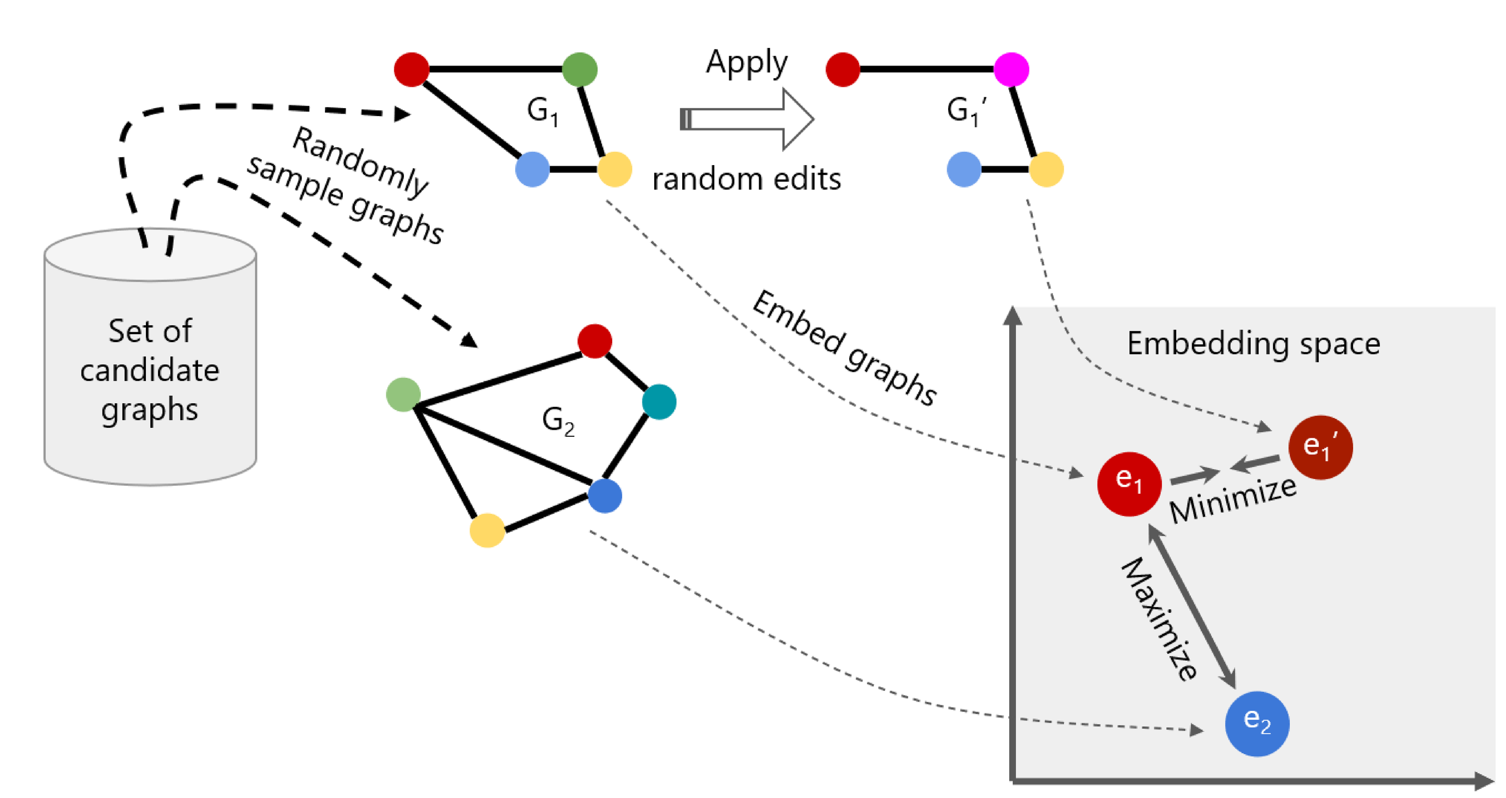

Figure 1: Our self-supervised task: Locality-based Classification.

- Why classify between edit-distance ≤ 6 vs. > 6?
  - NAS-Bench-101 [3] shows that architectures with low edit-distance have similar performance.
  - About 88% of the random graph pairs have edit-distance > 6.
  - Exact calculation of edit-distance is compute-intensive.
- Pretraining with various metric learning losses
  - We experiment with eight losses: Margin loss, Angular loss, NPairs loss, Multi-similarity loss, Lifted structure loss, Triplet margin loss, Contrastive loss, and Generalized lifted structure loss.

Applying to Neural Architecture Optimization [1]

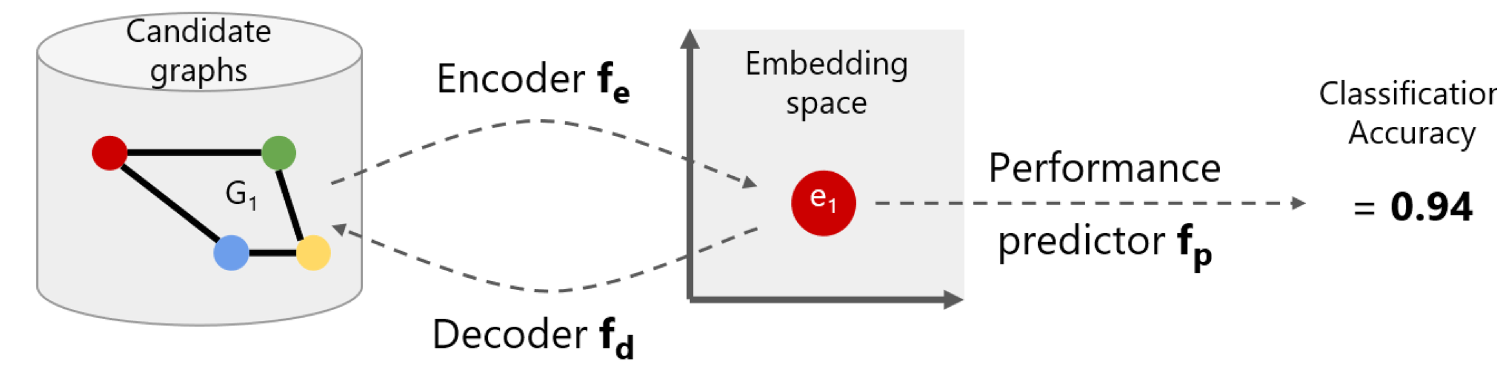

Figure 2: Components of Neural Architecture Optimization.

- Training loss of Neural Architecture Optimization

$$\mathcal{L}_{\text{Train}} = \mathcal{L}_{\text{Reconstruction}} + \mathcal{L}_{\text{Prediction}}$$
$$\mathcal{L}_{\text{Reconstruction}} = d(G, f_d(f_e(G)))$$
$$\mathcal{L}_{\text{Prediction}} = d(f_p(f_e(G)), a),$$

where (G, a) is the architecture-performance pair and $d(\cdot,\cdot)$ is the distance function.

- Our pretraining loss for the encoder and the decoder

$$\mathcal{L}_{\text{Pretrain}} = \mathcal{L}_{\text{Reconstruction}} + \mathcal{L}_{\text{Embedding}}$$
$$\mathcal{L}_{\text{Reconstruction}} = d(G, f_d(f_e(G)))$$
$$\mathcal{L}_{\text{Embedding}} = d(f_e(G_1), f_e(G_1')) - d(f_e(G_1), f_e(G_2)),$$

where (G_1, G_1') is the positive sample pair and (G_1, G_2) is the negative sample pair.

References

[1] Renqian Luo, Fei Tian, Tao Qin, Enhong Chen, and Tie-Yan Liu. Neural architecture optimization. In *Advances in neural information processing systems*, pages 7816--7827, 2018.

[2] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[3] Chris Ying, Aaron Klein, Eric Christiansen, Esteban Real, Kevin Murphy, and Frank Hutter. Nas-bench-101: Towards reproducible neural architecture search. In *International Conference on Machine Learning*, pages 7105--7114, 2019.

Experiments

- **Q. Can neural networks learn edit-distance? A. Yes.**

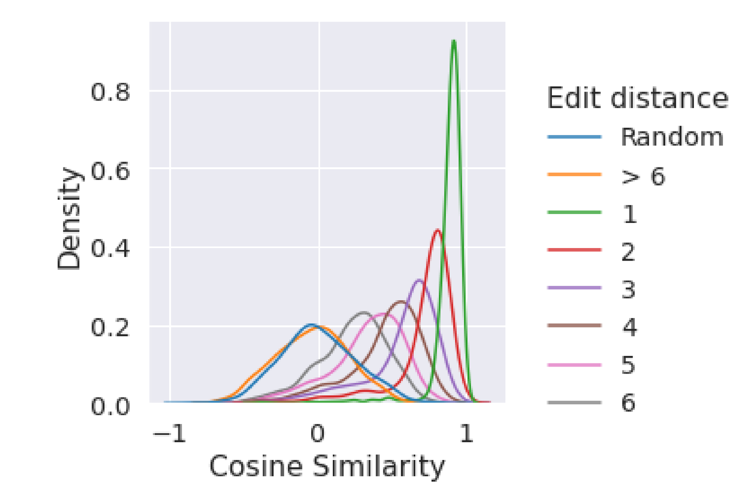

Figure 3: The graph pairs with lower edit-distance show smaller embedding distance, where the embeddings are obtained solely by solving the classification task between edit-distance > 6 vs. ≤ 6.

- **Q. Are embeddings preserved after training? A. Yes.**

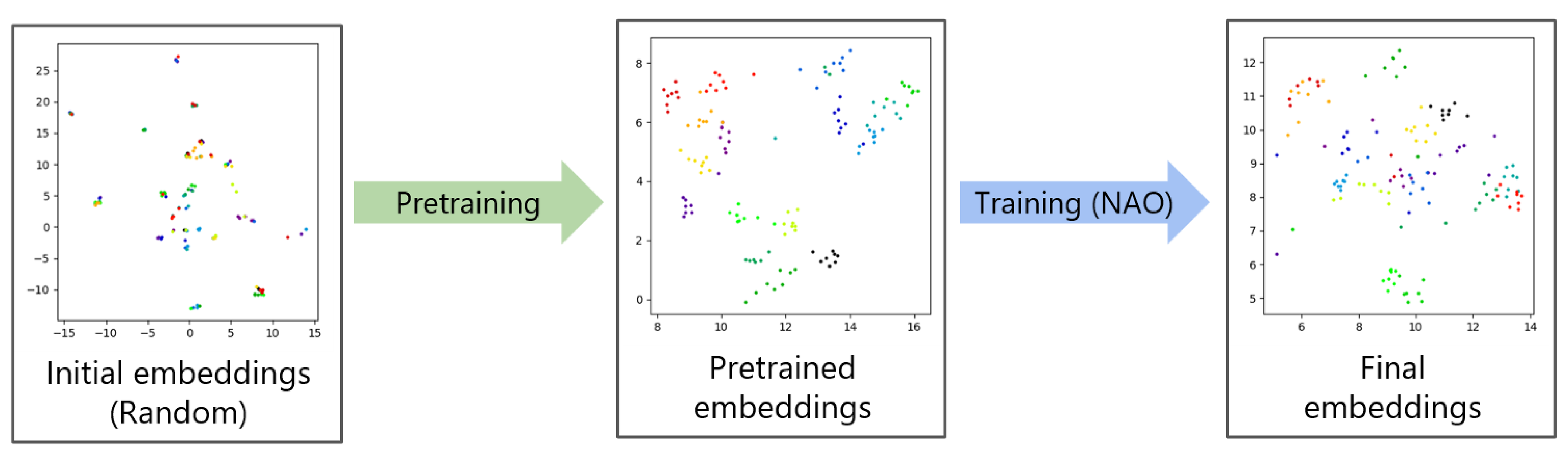

Figure 4: Visualization of the network embeddings using UMAP [2]. Same class samples have the same color.

- **Q. Is locality-based classification helpful? A. Sometimes.**

Method	Queries	Final Accuracy	Ranking
Using $\mathcal{L}_{\text{Pretraining}}$ (Best)	751	94.23	3
Only using $\mathcal{L}_{\text{Reconstruction}}$	810	94.23	2
Using $\mathcal{L}_{\text{Pretraining}}$ (Worst)	1007	94.22	3
Random baseline	560	93.84	336

Table 1: Average NAS performances of 5 repetitive runs, showing the best and worst metric learning method and two baseline methods. "Queries" is the number of architecture-accuracy pairs queried until NAO reaches the "Final accuracy". "Ranking" is the accuracy ranking of the model found among the whole NAS-Bench-101 dataset.