Design Space For Graph Neural Networks – 2020

Jiaxuan You, Rex Ying, Jure Leskovec

The Aim Of The Paper

New tasks are added to the field such as circuit design, sub graph matching, and more

which do not resemble existing benchmark tasks (node classification, …)

It is unclear how to design a new architecture suitable for that task

The Aim Of The Paper

Current research: design specific architectures, like: GCN, GIN, GAT

Their approach:

- A general design space
- A task space (with a similarity metric) for a novel task/data set
- Evaluation method

GOAL: Design well architectures and choose best out of them using task space

provides a systematical studying scheme for GNNs

GraphGym: a modularized pipeline consists of data/model/task/evaluation

tries to unify implementations available at: https://github.com/snap-stanford/GraphGym

Different from other methods

Vs architecture search:

Only focus on the design within layers/ evaluate on a small number of tasks (node classification)

Vs other evaluations:

They only focus on some specific designs

Vs transferable architecture search:

Transferring architecture search across multiple tasks

Assumption: different tasks follow the same distribution

Due to the great variety of graph learning, no longer holds

Terminology

Design = concrete GNN instantiation such as a 5-layer GraphSAGE

Each one can be characterized by multiple design dimensions

Experiment = applying a GNN design to a task

Which covers all combinations of designs and tasks

Design Space

Changing aggregation function to something like summation and adding skip connections create a new GNN model – A totally brand new GraphSage :)

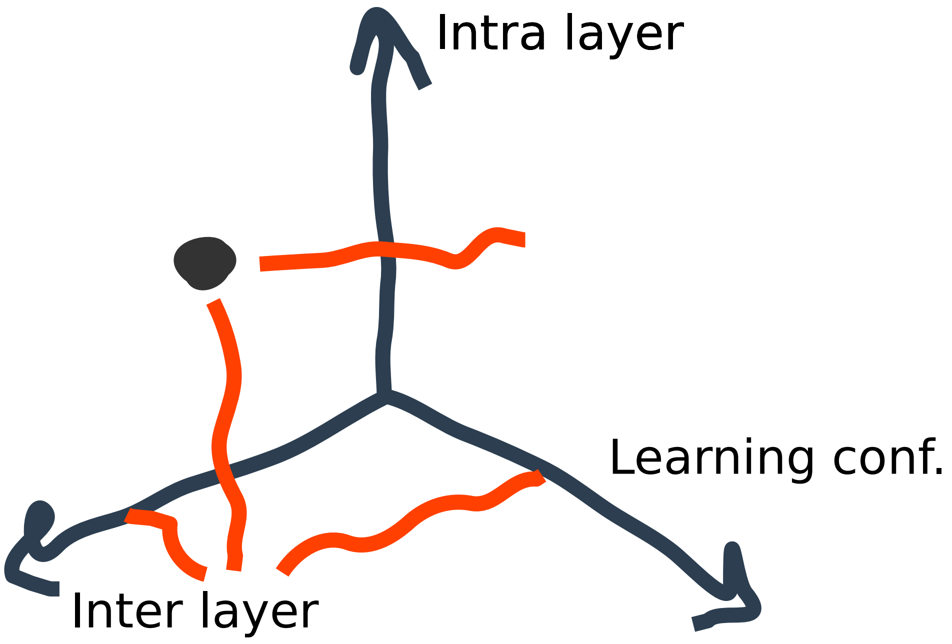
Consists of:

1. Intra-layer design

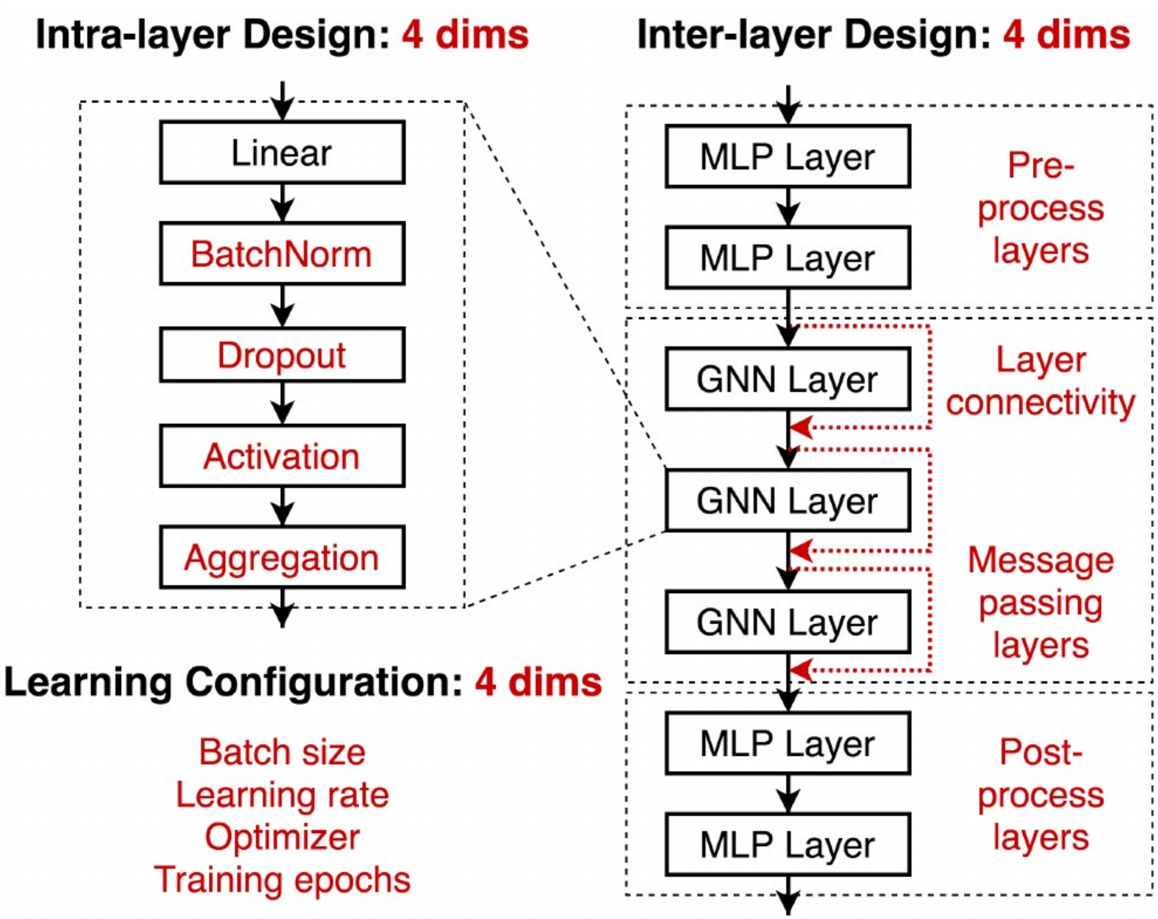
2. Inter-layer design

3. Learning configurations

12 dimensions, 31500 possibilities



Design Space



$$\mathbf{h}_v^{(k+1)} = \mathrm{AGG}\Big(\Big\{\mathrm{ACT}\Big(\mathrm{DROPOUT}\big(\mathrm{BN}(\mathbf{W}^{(k)}\mathbf{h}_u^{(k)} + \mathbf{b}^{(k)})\big)\Big), u \in \mathcal{N}(v)\Big\}\Big)$$

Task space

Goal: characterize similarities between different tasks

To identify a promising design for a new task/data set

Metric: Applying a fixed set of GNNs to two tasks and measuring Kendall rank correlation

Design space should be studied in conjunction with the task space

They consider 32 different tasks

Transfer best designs across tasks

Task space

How do we categorize different tasks to be representative of transferability of GNN designs?

Similarity metric between tasks:

1. Selection of anchor models

2. Rank distance of them

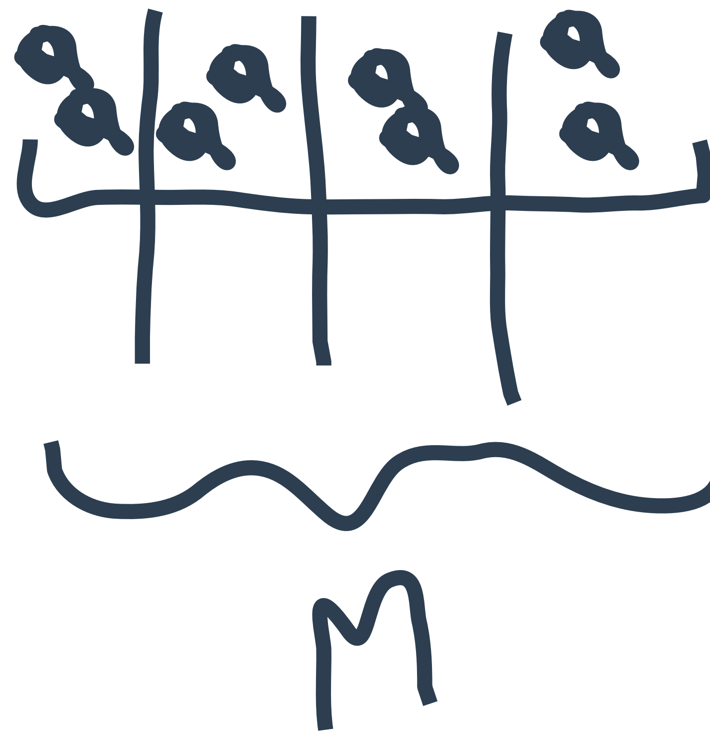
Task space

Selection of anchor models:

Choose D random GNN designs

Apply them to tasks

Choose median performance in each group



Task space

Measure task similarity:

Apply all M anchors to these tasks

Use Kendall rank correlation to compute the similarity between them

(Kendall's rank correlation measures the strength and direction of association that exists, between 1 and -1, uses pairs of observations,)

(M=12)

Concordant: (x1-x2)*(y1-y2) > 0, Discordant: other wise

$$\tau = \frac{(\text{number of concordant pairs}) - (\text{number of discordant pairs})}{\frac{1}{2}n(n-1)}.$$

Evaluation

Design and task space lead to over 10M possible combinations

So we need a way to evaluate our design space

Goal: Distilling the huge GNN design space by developing a controlled random search evaluation procedure

To BN, or not to BN, that is the question :)

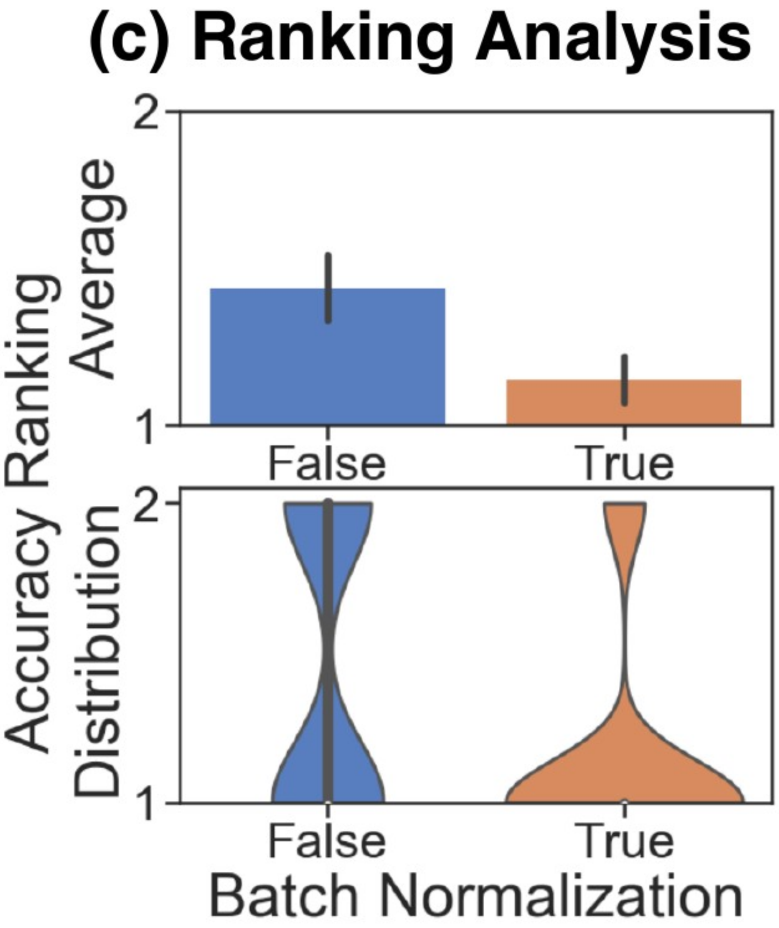
(a) Controlled Random Search

GNN Design Space					GNN Task Space	
BatchNorm	Activation	…	Message layers	Layer Connectivity	Task level	dataset
True	relu	…	8	skip_sum	node	CiteSeer
False	relu	…	8	skip_sum	node	CiteSeer
True	relu	…	2	skip_cat	graph	BZR
False	relu	…	2	skip_cat	graph	BZR
			...			
True	prelu	…	4	stack	graph	scale free
False	prelu	…	4	stack	graph	scale free

(b) Rank Design Choices by Performance

Experimental Results	
Val. Accuracy	Design Choice Ranking
0.75	1
0.54	2
0.88	1 (a tie)
0.86	1 (a tie)
0.89	1
0.36	2

Evaluation


(c) Ranking Analysis
Accuracy Ranking
Average
2
1
False
True
Distribution
2
1
False
True
Batch Normalization