WHAT GRAPH NEURAL NETWORKS CANNOT LEARN: DEPTH VS WIDTH

Paper Author: Andreas Loukas

Presented by: Hauton J Tsang

UNIVERSITY OF WATERLOO

Outline

- Introduction
- Background
- Importance of problem
- Limitations of previous methods
- Solutions of problem
- Interesting research questions

UNIVERSITY OF WATERLOO

INTRODUCTION

Introduction

- Expressivity of a machine learning model is important to know
- Universal approximation for neural networks
  - A large enough neural network can solve any problem
  - How useful is this information?
- What a model cannot learn may be more useful
  - Establish lower bounds for hyperparameters
  - Example: proving a problem cannot be solved with fewer than $f(n)$ layers for input size n

WHAT IS THE PROBLEM?

What is the problem?

- Authors want to analyze expressivity of message-passing GNNs (MPGNNs), particularly ones with node IDs
- Goals:
  - Formalize what problems MPGNNs can compute
  - Analyze what MPGNNs cannot compute under restrictions
    - Establish lower bounds for common problems

WHY IS IT IMPORTANT?

Why is it important?

- Formalizing what MPGNNs can compute:
  - Find blind spots of MPGNNs, if any
  - Provide a foundation to analyze limitations
- Formalizing what MPGNNs can't compute:
  - Theoretical lower bounds for hyperparameters for solving problems using GNNs
  - More informed design of MPGNN models

PRIOR WORK

Why don't previous methods address this problem?

- Small body of prior work on limitations in MPGNNs
  - Dehmamy et al. analyzed non-MPGNNs
  - Xu et al. and Morris et al. analyzed MPGNNs without node identification (ie. anonymous) using 1-WL
  - Sato et al. showed that partially-labelled MPGNNs are unable to approximate three NP-hard optimization problems well

Prior Work

- However, adding identifiers to nodes in a MPGNN significantly improves expressivity
- Identifiers can be added without violating permutation invariance/equivariance
- Authors analyze some problems that previous authors have not covered (decision, optimization, graph estimation)
- Depth and width of MPGNN directly connected with graph properties

UNIVERSITY OF WATERLOO

WHAT IS THE SOLUTION?

What can MPGNNs learn?

- Turns out, MPGNNs are Turing-universal
- Proof sketch:
  - MPGNNs have many similarities to LOCAL, a distributed computing model
  - Differences between MPGNNs and LOCAL:
    - MPGNNs must sum received messages before computing
    - Arguments of messaging function are different
    - Information representation is different

What can MPGNNs learn?

- However, the authors have proven that despite these differences, each node's computation in a MPGNN and LOCAL have the same expressivity
  - Given messaging and update functions are Turing-complete
  - For GNNs, this means that functions in each layer should be sufficiently complex

What can MPGNNs learn?

- LOCAL is Turing-complete if number of rounds of the distributed algorithm is larger than the graph diameter if memory is not an issue
- Thus, each GNN node can compute any Turing computable function if:
  - Depth (number of layers) d must be at least as great as the graph diameter
  - Width (largest state of node across all layers) w must be unbounded
- Since this result considers computation per node, the node must be uniquely identifiable for this result to hold

What can MPGNNs learn?

- Sufficient conditions for universality:
  - Uniquely identifiable node
  - Messaging and update functions must be sufficiently complex
  - Depth must be at least as large as graph diameter
  - Width must be unbounded

What can't MPGNNs learn?

- What happens if sufficient conditions for universality are relaxed?
- Analyze problems under constraints of depth and width of GNN
- It turns out restrictions on capacity significantly limit expressivity of MPGNNs
- Authors analyze relaxation of unique identification and depth/width conditions

What can't MPGNNs learn?

problem	*bound*	*problem*	*bound*
cycle detection (odd)	$dw = \Omega(n/\log n)$	shortest path	$d\sqrt{w} = \Omega(\sqrt{n}/\log n)$
cycle detection (even)	$dw = \Omega(\sqrt{n}/\log n)$	max. indep. set	$dw = \Omega(n^2/\log^2 n)$ for $w = O(1)$
subgraph verification*	$d\sqrt{w} = \Omega(\sqrt{n}/\log n)$	min. vertex cover	$dw = \Omega(n^2/\log^2 n)$ for $w = O(1)$
min. spanning tree	$d\sqrt{w} = \Omega(\sqrt{n}/\log n)$	perfect coloring	$dw = \Omega(n^2/\log^2 n)$ for $w = O(1)$
min. cut	$d\sqrt{w} = \Omega(\sqrt{n}/\log n)$	girth 2-approx.	$dw = \Omega(\sqrt{n}/\log n)$
diam. computation	$dw = \Omega(n/\log n)$	diam. $3/2$-approx.	$dw = \Omega(\sqrt{n}/\log n)$

- where d=depth, w=width, n=# of nodes in the graph

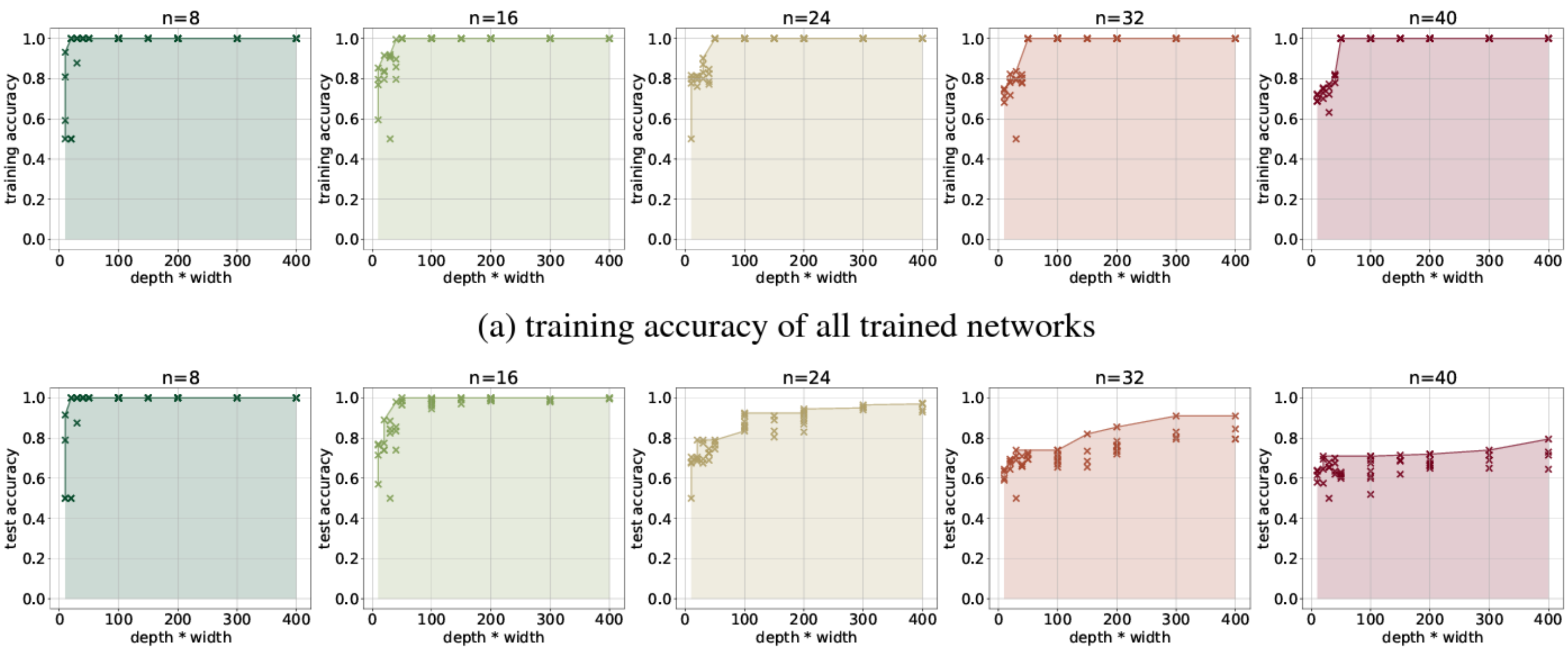
UNIVERSITY OF WATERLOO

What can't MPGNNs learn?

- Proof sketch:
  - Show equivalence of expressivity limits between MPGNN and CONGEST, a variant of LOCAL
  - For each problem, find equivalent in CONGEST
  - Established limits of expressivity of CONGEST can be translated into limits of MPGNN

What can't MPGNNs learn?

- Problem: 4-cycle classification
- Empirical results with constrained capacity *dw*



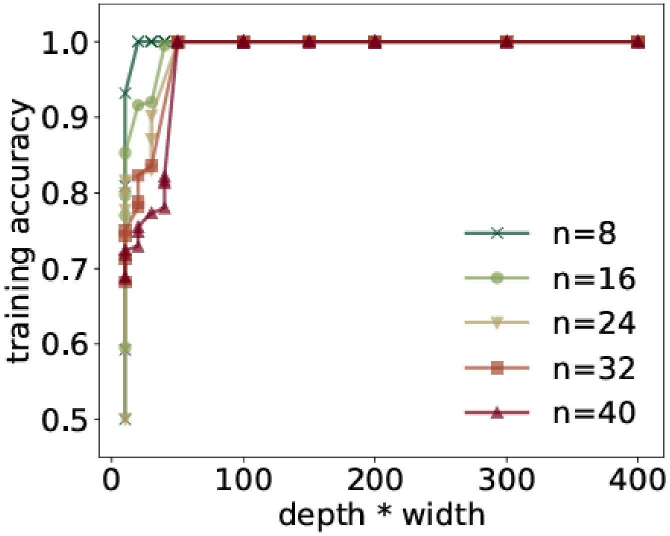
(a) training accuracy of all trained networks

(b) test accuracy of all trained networks

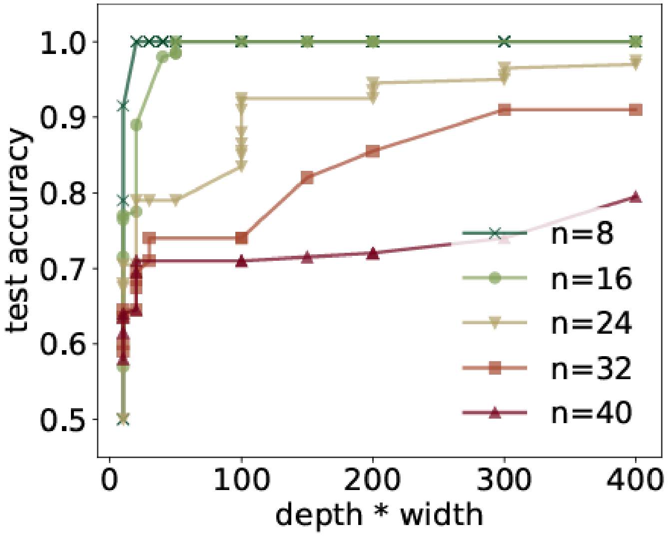
UNIVERSITY OF WATERLOO

What can’t MPGNNs learn?

- Empirical results with constrained capacity dw



(c) best training accuracy

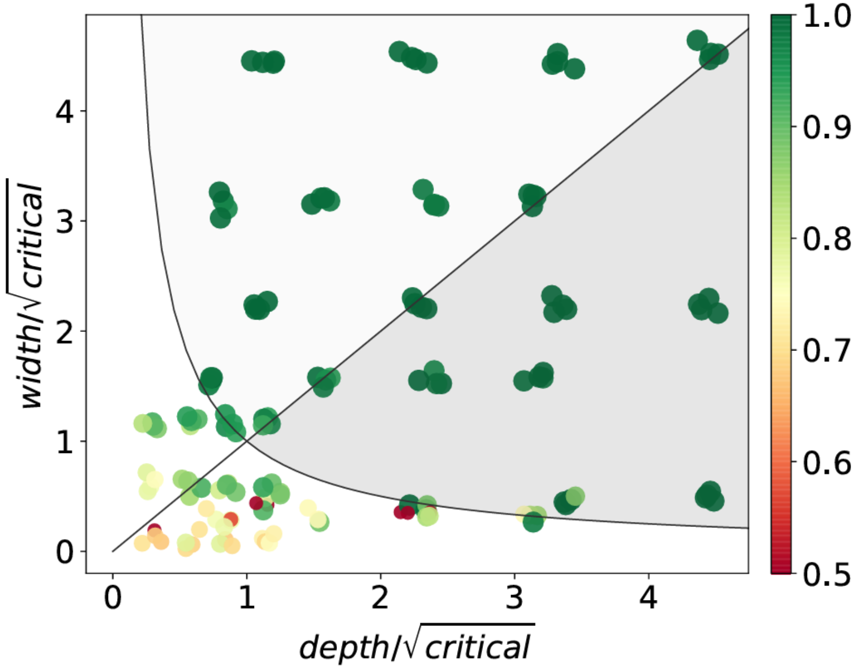


(d) best test accuracy

UNIVERSITY OF WATERLOO

What can't MPGNNs learn?

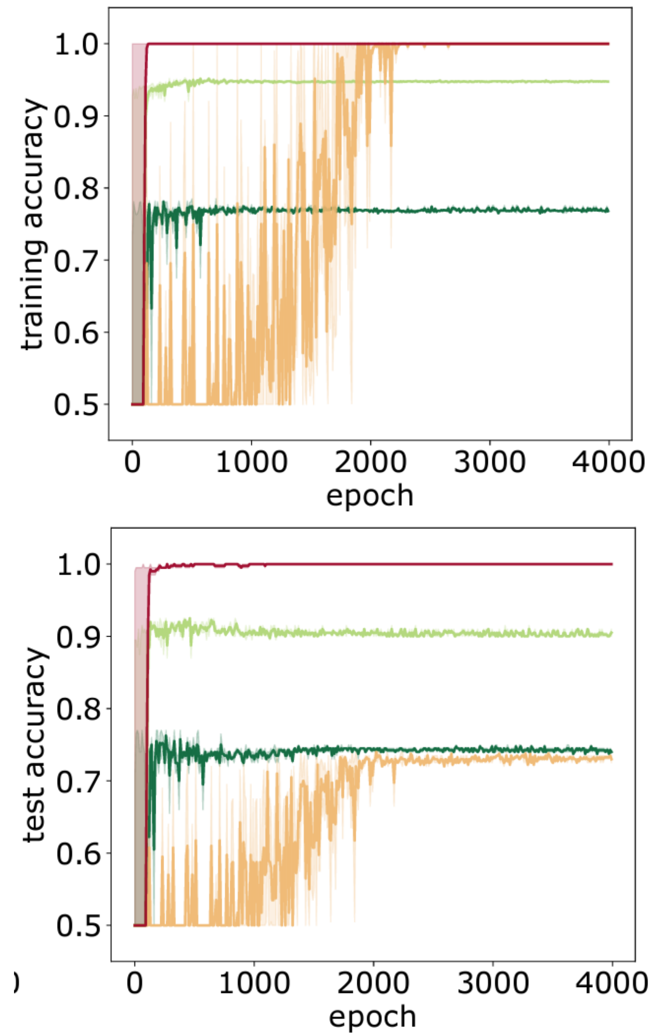
- Empirical results with independently constrained depth d and width w



UNIVERSITY OF WATERLOO

What can’t MPGNNs learn?

- Empirical results with various levels of anonymity:
  - Anonymous: no graph IDs
  - Degree: ID is node degree
  - Random unique ID: Inconsistent node ID between graphs
  - Unique ID: consistent node ID between graphs



Limitations of results

- Analyzed lower bounds are worst-case
  - One impossible graph is enough to prove lower bound
  - Does not mean they are all impossible
- Lower bounds were found with the assumption of universal layers and nodes with discriminative attributes
- However, lower bound results will still be applicable for graphs with less expressivity (e.g. computationally limited layers, anonymous graphs)

FUTURE DIRECTIONS

Future Directions

- Non-worst case bounds?
  - How tight are the derived bounds?
- Loosening assumptions on complexity of message and update functions