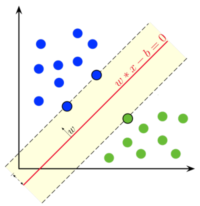
Two Analyses of Modern Deep Learning: Graph Neural Networks and Language Model Finetuning

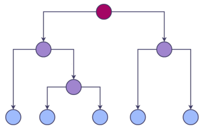
Noam Razin
Tel Aviv University

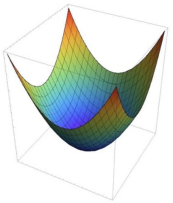
Machine Learning Paradigms

Machine Learning Paradigms

Classical Machine Learning





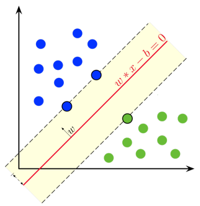
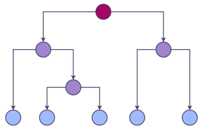
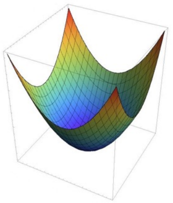


Models: Linear predictors, decision trees,...

Typical Properties: Convex, underparameterized

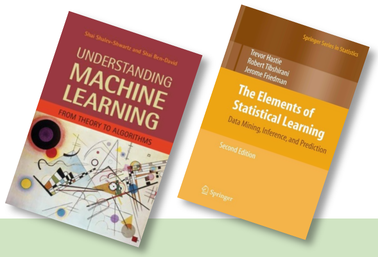
Machine Learning Paradigms

Classical Machine Learning

Models: Linear predictors, decision trees,...

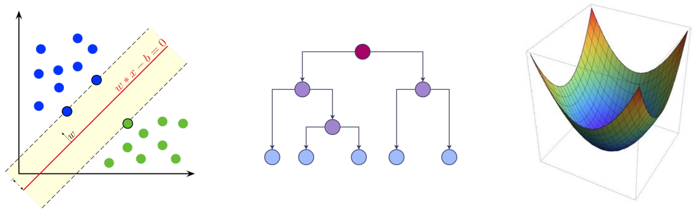
Typical Properties: Convex, underparameterized



Theory: Well-established

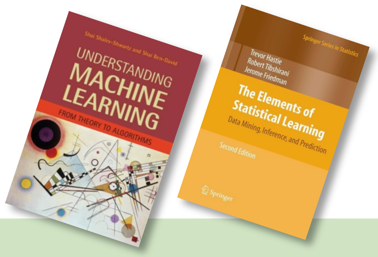
Machine Learning Paradigms

Classical Machine Learning



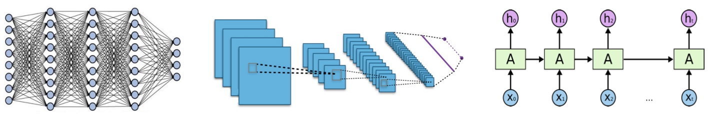
Models: Linear predictors, decision trees,…

Typical Properties: Convex, underparameterized



Theory: Well-established

"Classical" Deep Learning

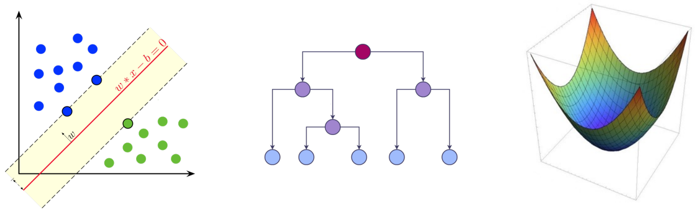


Models: Fully-Connected NN, CNN, RNN

Typical Properties: Non-convex, overparameterized, supervised learning

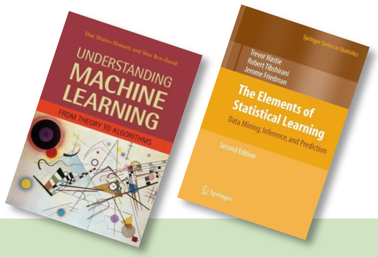
Machine Learning Paradigms

Classical Machine Learning



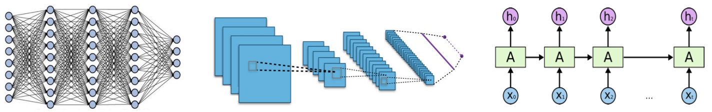
Models: Linear predictors, decision trees,…

Typical Properties: Convex, underparameterized



Theory: Well-established

"Classical" Deep Learning



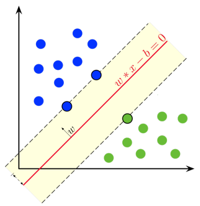
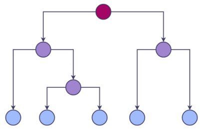
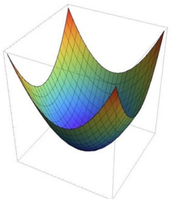
Models: Fully-Connected NN, CNN, RNN

Typical Properties: Non-convex, overparameterized, supervised learning

Theory: In progress

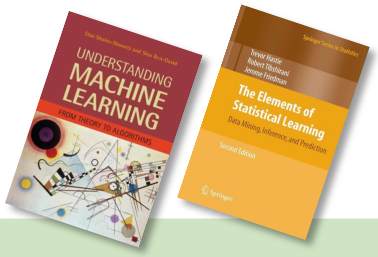
Machine Learning Paradigms

Classical Machine Learning

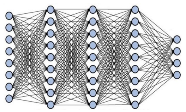
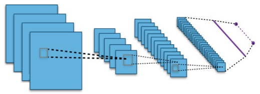
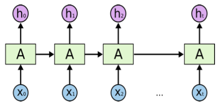
Models: Linear predictors, decision trees,...

Typical Properties: Convex, underparameterized



Theory: Well-established

"Classical" Deep Learning

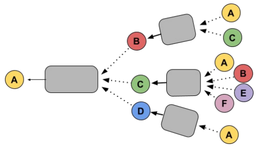
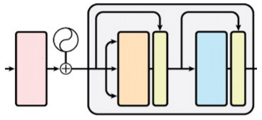




Models: Fully-Connected NN, CNN, RNN

Typical Properties: Non-convex, overparameterized, supervised learning

Theory: In progress

Modern Deep Learning

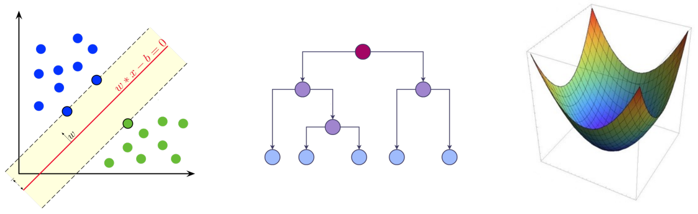




Models: GNN, Transformer, State Space Model,...

Typical Properties: Self-supervised foundation models, finetuning, underparameterized

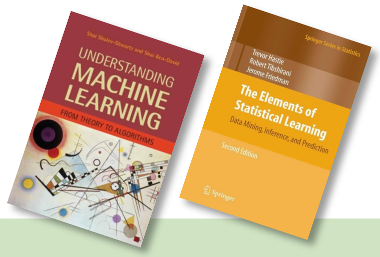
Machine Learning Paradigms

Classical Machine Learning



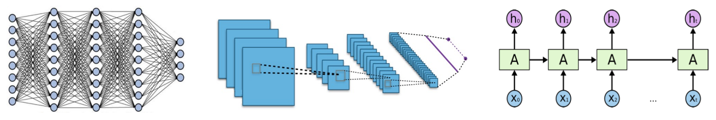
Models: Linear predictors, decision trees,...

Typical Properties: Convex, underparameterized



Theory: Well-established

"Classical" Deep Learning

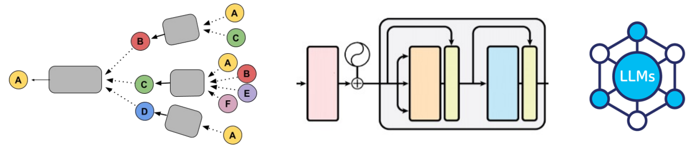


Models: Fully-Connected NN, CNN, RNN

Typical Properties: Non-convex, overparameterized, supervised learning

Theory: In progress

Modern Deep Learning



Models: GNN, Transformer, State Space Model,...

Typical Properties: Self-supervised foundation models, finetuning, underparameterized

Theory: Limited

My Research: Theoretical Foundations of Deep Learning

My Research: Theoretical Foundations of Deep Learning

"Classical" Deep Learning

Implicit Regularization in Deep Learning May Not Be Explainable by Norms

R + Cohen | *NeurIPS 2020*

Implicit Regularization in Tensor Factorization

R + Maman + Cohen | *ICML 2021*

Implicit Regularization in Hierarchical Tensor Factorization and Deep Convolutional Neural Networks

R + Maman + Cohen | *ICML 2022*

What Makes Data Suitable for a Locally Connected Neural Network? A Necessary and Sufficient Condition Based on Quantum Entanglement

Alexander + De La Vega + **R** + Cohen | *NeurIPS 2023*

Modern Deep Learning

On the Ability of Graph Neural Networks to Model Interactions Between Vertices

R + Verbin + Cohen | *NeurIPS 2023*

Vanishing Gradients in Reinforcement Finetuning of Language Models

R + Zhou + Saremi + Thilak + Bradley + Nakkiran + Susskind + Littwin | *arXiv*

What Algorithms Can Transformers Learn? A Study in Length Generalization

Zhou + Bradley + Littwin + **R** + Saremi + Susskind + Bengio + Nakkiran | *arXiv*

My Research: Theoretical Foundations of Deep Learning

"Classical" Deep Learning

Implicit Regularization in Deep Learning May Not Be Explainable by Norms

R + Cohen | *NeurIPS 2020*

Implicit Regularization in Tensor Factorization

R + M

Generalization and suitability of data to deep learning via dynamical analyses and connections to tensor factorizations

Hierarchical Tensor Factorization and Deep Convolutional Neural Networks

R + Maman + Cohen | *ICML 2022*

What Makes Data Suitable for a Locally Connected Neural Network? A Necessary and Sufficient Condition Based on Quantum Entanglement

Alexander + De La Vega + *R* + Cohen | *NeurIPS 2023*

Modern Deep Learning

On the Ability of Graph Neural Networks to Model Interactions Between Vertices

R + Verbin + Cohen | *NeurIPS 2023*

Vanishing Gradients in Reinforcement Finetuning of Language Models

R + Zhou + Saremi + Thilak + Bradley + Nakkiran + Susskind + Littwin | *arXiv*

What Algorithms Can Transformers Learn? A Study in Length Generalization

Zhou + Bradley + Littwin + *R* + Saremi + Susskind + Bengio + Nakkiran | *arXiv*

My Research: Theoretical Foundations of Deep Learning

"Classical" Deep Learning

Implicit Regularization in Deep Learning May Not Be Explainable by Norms

R + Cohen | *NeurIPS 2020*

Implicit Regularization in Tensor Factorization

R + M...

Generalization and suitability of data to deep learning via dynamical analyses and connections to tensor factorizations

...Hierarchical Tensor Factorization and Deep Convolutional Neural Networks

R + Maman + Cohen | *ICML 2022*

What Makes Data Suitable for a Locally Connected Neural Network? A Necessary and Sufficient Condition Based on Quantum Entanglement

Alexander + De La Vega + *R* + Cohen | *NeurIPS 2023*

Modern Deep Learning

On the Ability of Graph Neural Networks to Model Interactions Between Vertices

R + Verbin + Cohen | *NeurIPS 2023*

Vanishing Gradients in Reinforcement Finetuning of Language Models

R + Zhou + Saremi + Thilak + Bradley + Nakkiran + Susskind + Littwin | *arXiv*

What Algorithms Can Transformers Learn? A Study in Length Generalization

Zhou + Bradley + Littwin + *R* + Saremi + Susskind + Bengio + Nakkiran | *arXiv*

On the Ability of Graph Neural Networks to Model Interactions Between Vertices

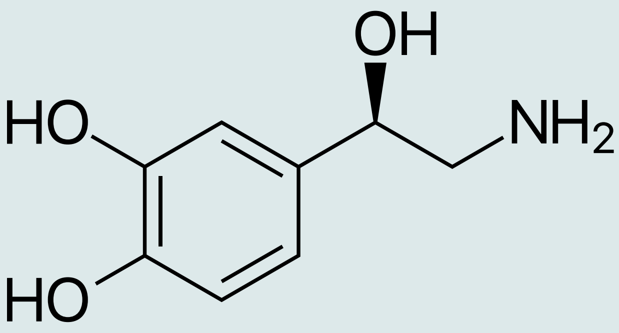
Graph Neural Networks (GNNs)

Neural networks purposed for **modeling interactions over graph data**

Graph Neural Networks (GNNs)

Neural networks purposed for **modeling interactions over graph data**

Graph Prediction



Vertex Prediction

Expressivity of GNNs

Challenge

Develop mathematical theory for GNNs

Expressivity of GNNs

Challenge

Develop mathematical theory for GNNs

Fundamental Question

Expressivity: Which functions can GNNs realize?

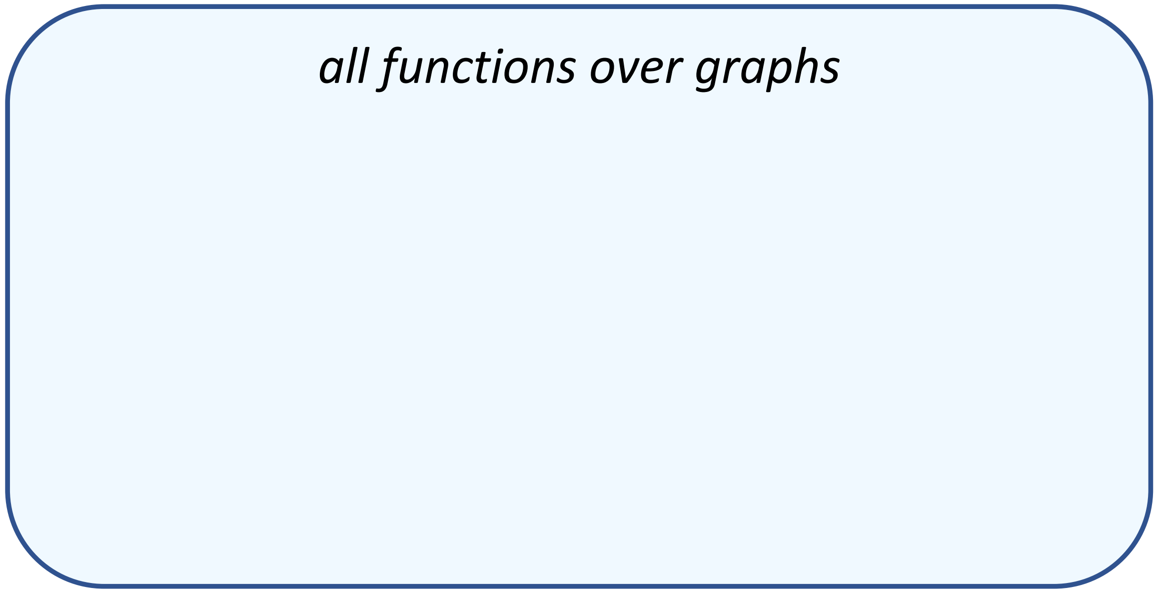
Expressivity of GNNs

Challenge

Develop mathematical theory for GNNs

Fundamental Question

Expressivity: Which functions can GNNs realize?



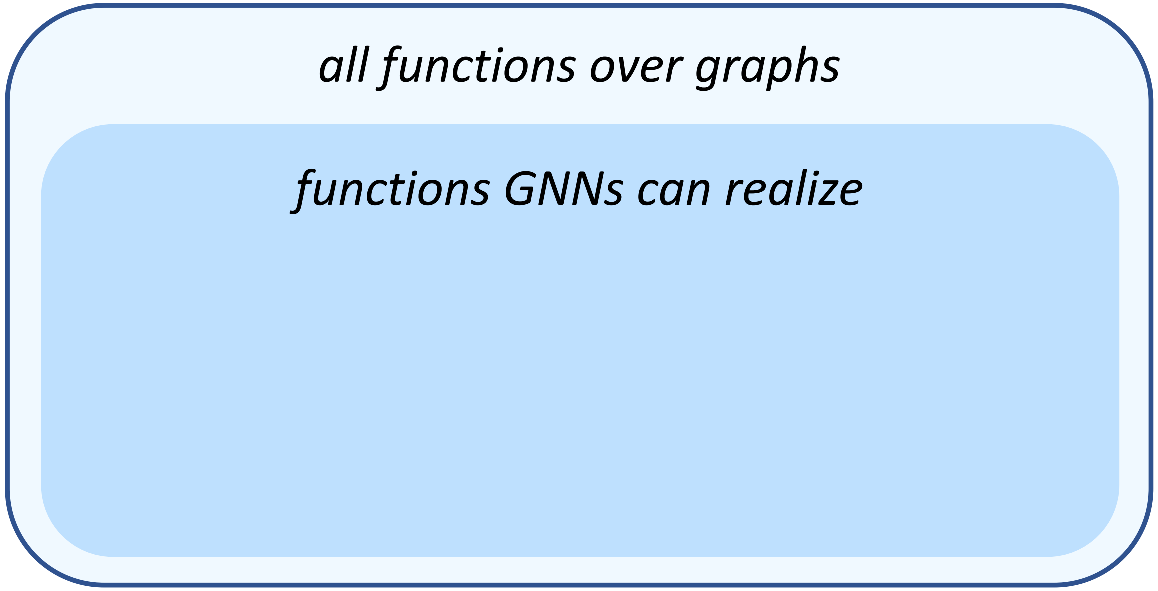
Expressivity of GNNs

Challenge

Develop mathematical theory for GNNs

Fundamental Question

Expressivity: Which functions can GNNs realize?



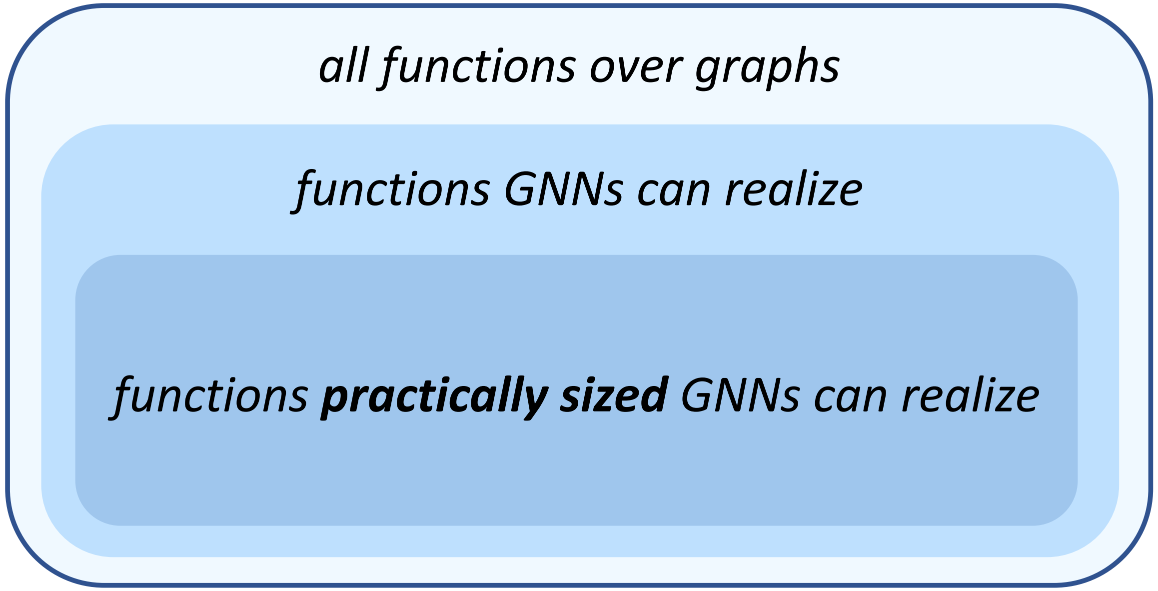
Expressivity of GNNs

Challenge

Develop mathematical theory for GNNs

Fundamental Question

Expressivity: Which functions can GNNs realize?



Limitations of Existing Analyses

Theoretical analysis of GNN expressivity is an active area

Limitations of Existing Analyses

Theoretical analysis of GNN expressivity is an active area

(e.g. Xu et al. 2019, Morris et al. 2019, Maron et al. 2019a, Maron et al. 2019b, Keriven & Peyré 2019, Chen et al. 2019, Dehmamy et al. 2019, Garg et al. 2020, Loukas 2020, Chen et al. 2020, Azizian & Lelarge 2021, Geerts & Reutter 2022, Zhang et al. 2023)

Limitations of Existing Analyses

Theoretical analysis of GNN expressivity is an active area

(e.g. Xu et al. 2019, Morris et al. 2019, Maron et al. 2019a, Maron et al. 2019b, Keriven & Peyré 2019, Chen et al. 2019, Dehmamy et al. 2019, Garg et al. 2020, Loukas 2020, Chen et al. 2020, Azizian & Lelarge 2021, Geerts & Reutter 2022, Zhang et al. 2023)

Limitations: Despite recent progress, existing analyses

Limitations of Existing Analyses

Theoretical analysis of GNN expressivity is an active area

(e.g. Xu et al. 2019, Morris et al. 2019, Maron et al. 2019a, Maron et al. 2019b, Keriven & Peyré 2019, Chen et al. 2019, Dehmamy et al. 2019, Garg et al. 2020, Loukas 2020, Chen et al. 2020, Azizian & Lelarge 2021, Geerts & Reutter 2022, Zhang et al. 2023)

Limitations: Despite recent progress, existing analyses

(1) Often treat regimes of **unbounded width or depth**

Limitations of Existing Analyses

Theoretical analysis of GNN expressivity is an active area

(e.g. Xu et al. 2019, Morris et al. 2019, Maron et al. 2019a, Maron et al. 2019b, Keriven & Peyré 2019, Chen et al. 2019, Dehmamy et al. 2019, Garg et al. 2020, Loukas 2020, Chen et al. 2020, Azizian & Lelarge 2021, Geerts & Reutter 2022, Zhang et al. 2023)

Limitations: Despite recent progress, existing analyses

(1) Often treat regimes of **unbounded width or depth**

(2) Do not formalize ability to **model interactions between vertices**

Limitations of Existing Analyses

Theoretical analysis of GNN expressivity is an active area

(e.g. Xu et al. 2019, Morris et al. 2019, Maron et al. 2019a, Maron et al. 2019b, Keriven & Peyré 2019, Chen et al. 2019, Dehmamy et al. 2019, Garg et al. 2020, Loukas 2020, Chen et al. 2020, Azizian & Lelarge 2021, Geerts & Reutter 2022, Zhang et al. 2023)

Limitations: Despite recent progress, existing analyses

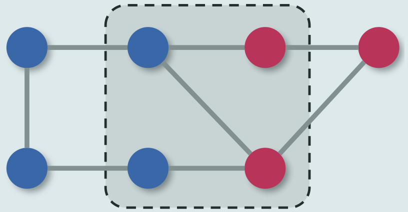
(1) Often treat regimes of **unbounded width or depth**

(2) Do not formalize ability to **model interactions between vertices**

Q: How do graph structure and GNN architecture affect interactions?

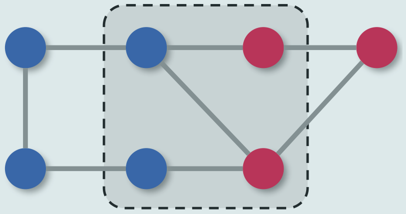
Main Contributions: Ability of GNNs to Model Interactions

Main Contributions: Ability of GNNs to Model Interactions

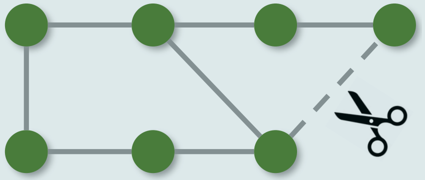


Theory: Characterize ability of certain GNNs to model interactions between vertices

Main Contributions: Ability of GNNs to Model Interactions

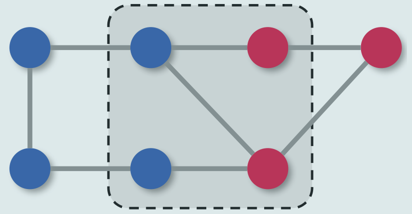


Theory: Characterize ability of certain GNNs to **model interactions between vertices**

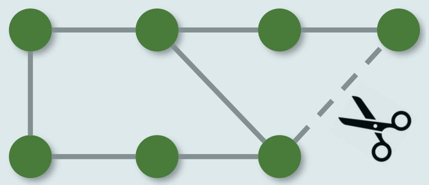


Application: **Edge sparsification** algorithm preserving interactions

Main Contributions: Ability of GNNs to Model Interactions

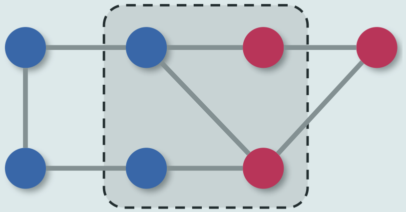


Theory: Characterize ability of certain GNNs to model interactions between vertices

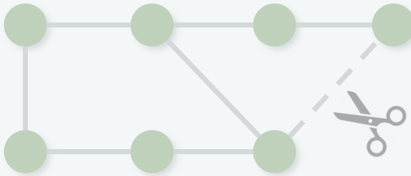


Application: Edge sparsification algorithm preserving interactions

Main Contributions: Ability of GNNs to Model Interactions



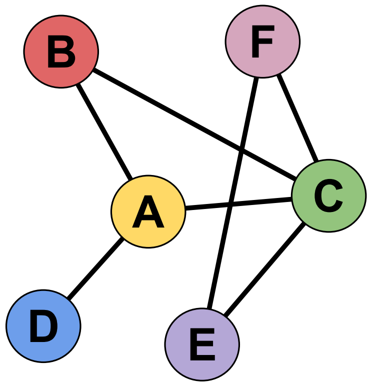
Theory: Characterize ability of certain GNNs to model interactions between vertices



Application: Edge sparsification algorithm preserving interactions

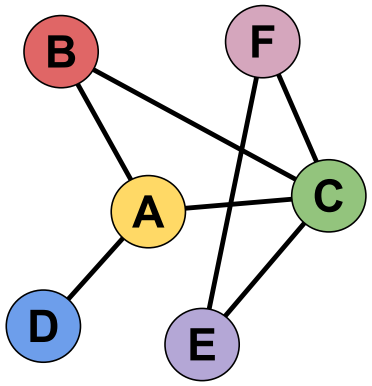
Message-Passing GNNs

Message-Passing GNNs



Inputs: Graph $G = (V, E)$, vertex features $X = \left(x^{(1)}, \ldots, x^{(|V|)}\right)$

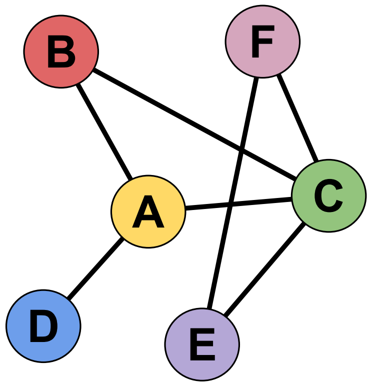
Message-Passing GNNs



Inputs: Graph $G = (V, E)$, vertex features $X = \left(x^{(1)}, \ldots, x^{(|V|)}\right)$

Initialize: $h^{(0,i)} := x^{(i)}$ for $i \in V$

Message-Passing GNNs

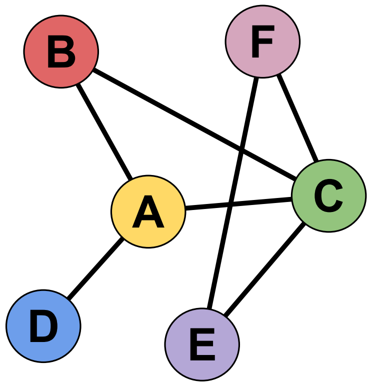


Inputs: Graph $G = (V, E)$, vertex features $X = \left(x^{(1)}, \ldots, x^{(|V|)}\right)$

Initialize: $h^{(0,i)} := x^{(i)}$ for $i \in V$

Common Update Rule: At layer $l = 1, \ldots, L$ for $i \in V$

Message-Passing GNNs



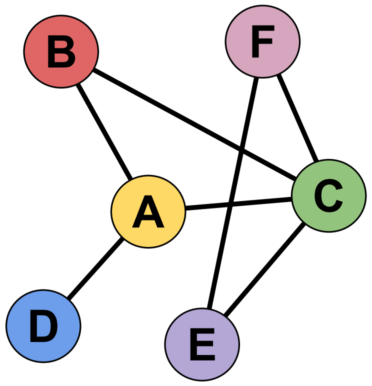
Inputs: Graph $G = (V, E)$, vertex features $X = \left(x^{(1)}, \ldots, x^{(|V|)}\right)$

Initialize: $h^{(0,i)} := x^{(i)}$ for $i \in V$

Common Update Rule: At layer $l = 1, \ldots, L$ for $i \in V$

$$h^{(l,i)} = \text{AGG}\left(\left\{W^{(l)} h^{(l-1,j)} : j \in \text{neighbors}(i)\right\}\right)$$

Message-Passing GNNs



Inputs: Graph $G = (V, E)$, vertex features $X = \left(x^{(1)}, \ldots, x^{(|V|)}\right)$

Initialize: $h^{(0,i)} := x^{(i)}$ for $i \in V$

Common Update Rule: At layer $l = 1, \ldots, L$ for $i \in V$

$$h^{(l,i)} = \textsc{prod}\left(\left\{W^{(l)} h^{(l-1,j)} : j \in \text{neighbors}(i)\right\}\right)$$

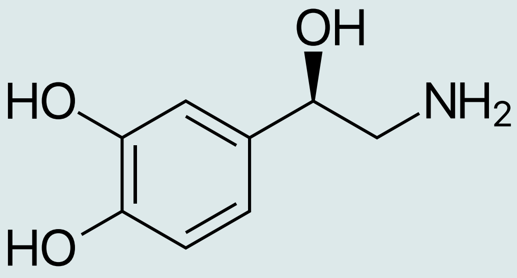
GNNs for Vertex vs Graph Prediction

After L layers the GNN produces $h^{(L,1)}, \ldots, h^{(L,|V|)}$

GNNs for Vertex vs Graph Prediction

After L layers the GNN produces $h^{(L,1)}, \ldots, h^{(L,|V|)}$

Graph Prediction: Single output for the whole graph

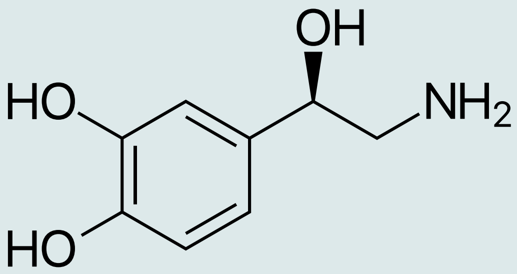


$$GNN(X) = W^{(o)} \mathrm{AGG}\left(h^{(L,1)}, \ldots, h^{(L,|V|)}\right)$$

GNNs for Vertex vs Graph Prediction

After L layers the GNN produces $h^{(L,1)}, \ldots, h^{(L,|V|)}$

Graph Prediction: Single output for the whole graph



$$GNN(X) = W^{(o)} \text{AGG}\left(h^{(L,1)}, \ldots, h^{(L,|V|)}\right)$$

Vertex Prediction: Output for every $t \in V$

$$GNN^{(t)}(X) = W^{(o)} h^{(L,t)}$$

Separation Rank

Separation Rank

Widely used measure for **interaction modeled across partition of input variables**

Separation Rank

Widely used measure for **interaction modeled across partition of input variables**

(input variables: vertices of an input graph)

Separation Rank

Widely used measure for **interaction modeled across partition of input variables**

vertices of an input graph

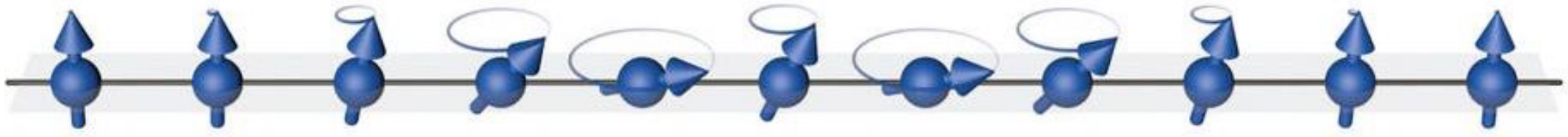
- Measure of **entanglement** in quantum mechanics

Separation Rank

Widely used measure for **interaction modeled across partition of input variables**

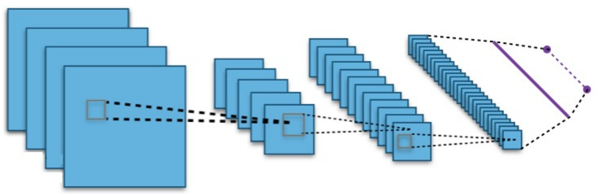
vertices of an input graph

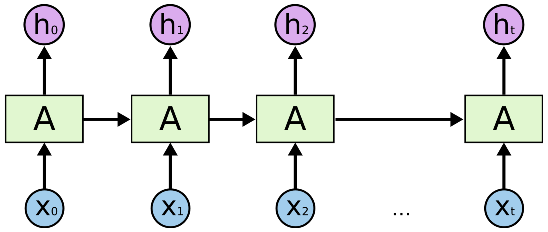
- Measure of **entanglement** in quantum mechanics

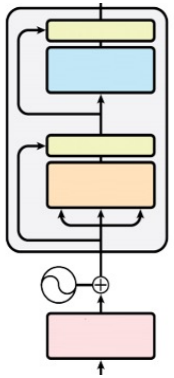


- Analyses of convolutional, recurrent, and self-attention NNs

  (e.g. Cohen & Shashua 2017, Levine et al. 2018;2020, **R** et al. 2022)



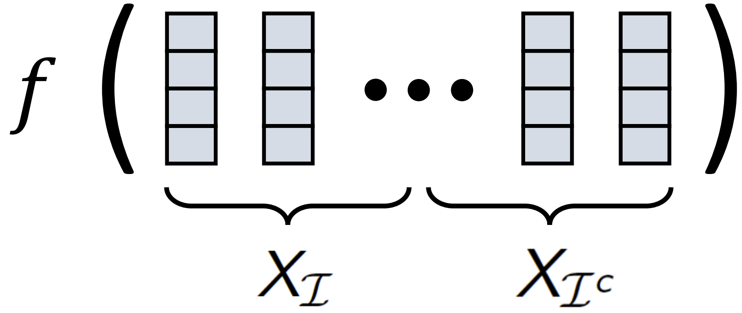




Separation Rank: Formal Definition

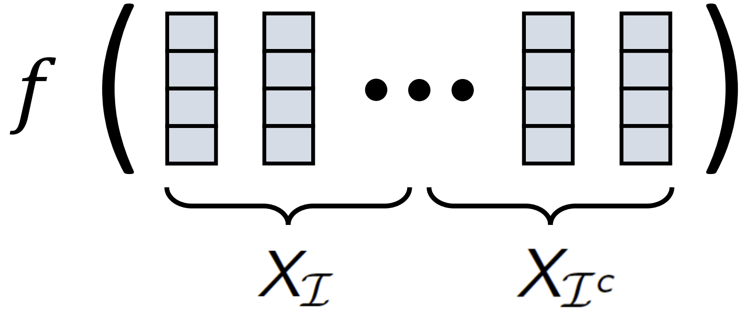
Separation Rank: Formal Definition

Let $f : (\mathbb{R}^D)^N \to \mathbb{R}$ and subset of variables $\mathcal{I} \subseteq \{1, \ldots, N\}$



Separation Rank: Formal Definition

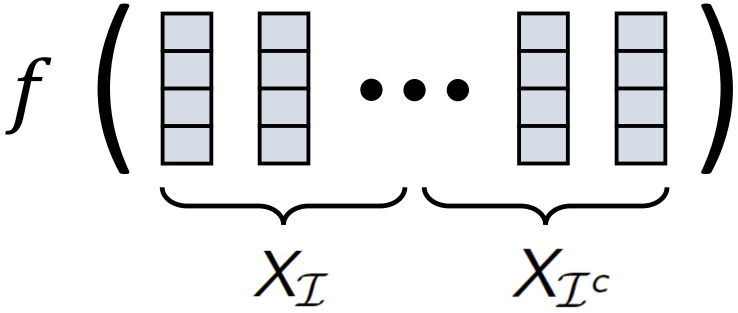
Let $f : (\mathbb{R}^D)^N \to \mathbb{R}$ and subset of variables $\mathcal{I} \subseteq \{1, \ldots, N\}$



$$\mathrm{sep}(f;\mathcal{I}) := \min\ R \text{ s.t. } f(X) = \sum\nolimits_{r=1}^{R} g_r(X_{\mathcal{I}}) \cdot \bar{g}_r(X_{\mathcal{I}^c})$$

Separation Rank: Formal Definition

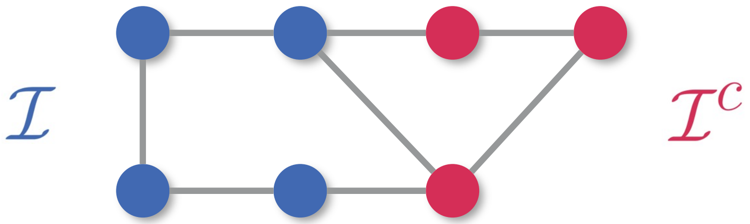
Let $f : (\mathbb{R}^D)^N \to \mathbb{R}$ and subset of variables $\mathcal{I} \subseteq \{1, \ldots, N\}$



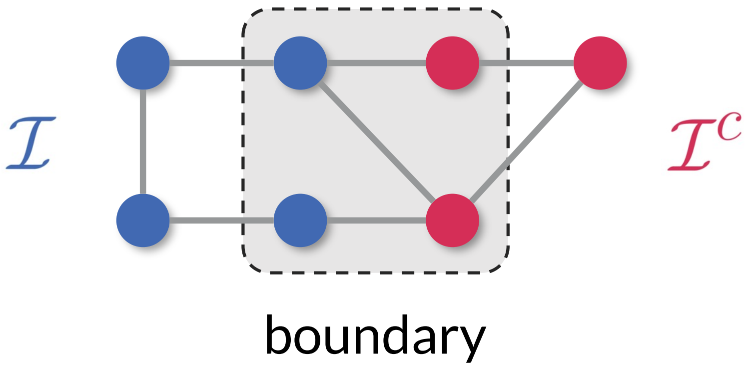
$$\mathrm{sep}(f;\mathcal{I}) := \min\ R \text{ s.t. } f(X) = \sum_{r=1}^{R} g_r(X_{\mathcal{I}}) \cdot \bar{g}_r(X_{\mathcal{I}^c})$$

Higher $\mathrm{sep}(f;\mathcal{I})$ ⟶ stronger interaction between $X_{\mathcal{I}}$ and $X_{\mathcal{I}^c}$

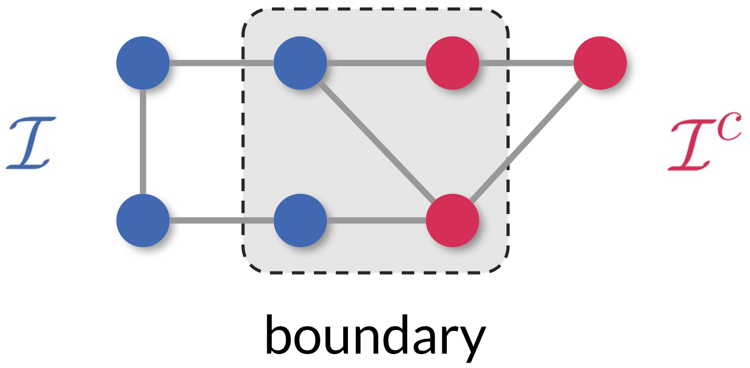
Definition: Walk Index (WI) of a Partition of Vertices



Definition: Walk Index (WI) of a Partition of Vertices



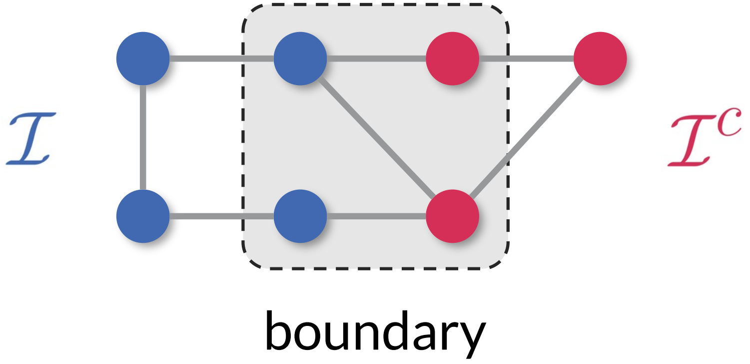
Definition: Walk Index (WI) of a Partition of Vertices

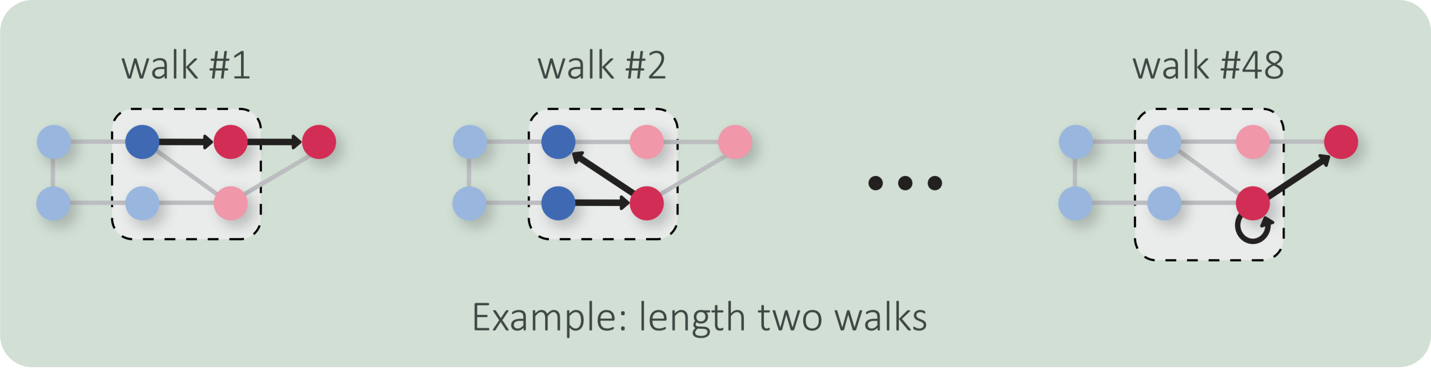


Graph Prediction (with depth L GNN)

$$\mathrm{WI}_{L-1}(\mathcal{I}) := \text{\# length } L-1 \text{ walks from boundary}$$

Definition: Walk Index (WI) of a Partition of Vertices

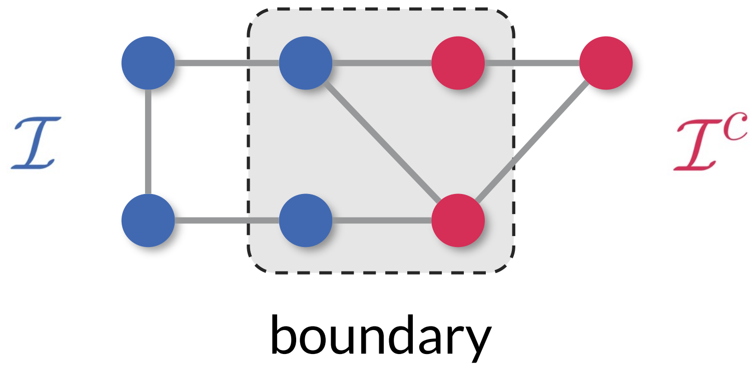


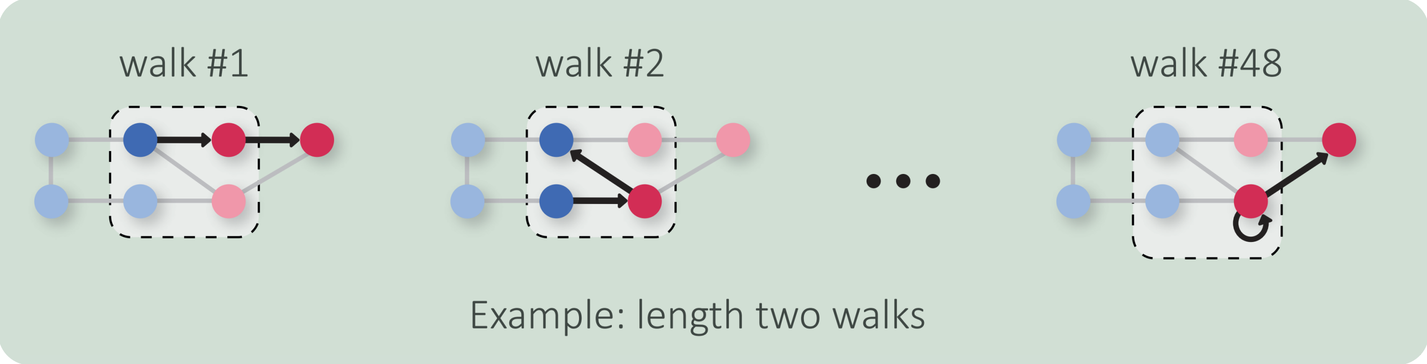


Graph Prediction (with depth L GNN)

$$\mathrm{WI}_{L-1}(\mathcal{I}) := \text{\# length } L-1 \text{ walks from boundary}$$

Definition: Walk Index (WI) of a Partition of Vertices





Graph Prediction (with depth L GNN)

$$\mathrm{WI}_{L-1}(\mathcal{I}) := \text{\# length } L-1 \text{ walks from boundary}$$

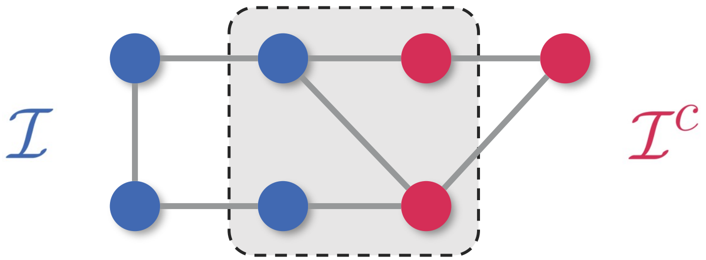
Vertex Prediction (with depth L GNN)

$$\mathrm{WI}_{L-1,t}(\mathcal{I}) := \text{\# length } L-1 \text{ walks from boundary to } t \in V$$

Main Result: Strength of Interaction $\propto$ Walk Index

Theorem

For a depth L GNN with width D and $\mathcal{I} \subseteq V$:

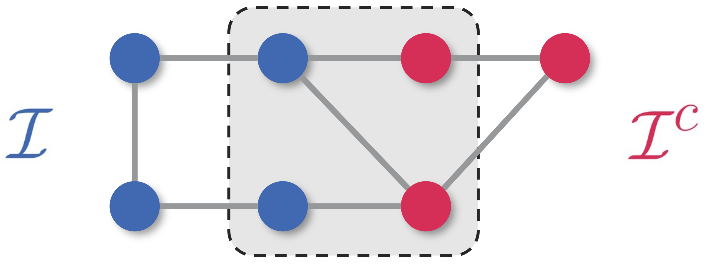


Main Result: Strength of Interaction ∝ Walk Index

Theorem

For a depth L GNN with width D and $\mathcal{I} \subseteq V$:

(graph prediction) $$\mathrm{sep}(GNN; \mathcal{I}) = D^{\mathcal{O}(\mathbf{WI}_{L-1}(\mathcal{I}))}$$



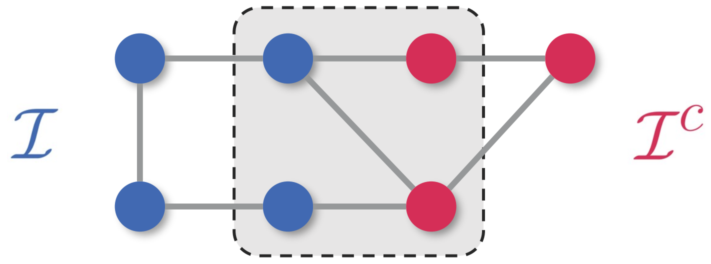
Main Result: Strength of Interaction ∝ Walk Index

Theorem

For a depth L GNN with width D and $\mathcal{I} \subseteq V$:

(graph prediction) $\quad \mathrm{sep}(GNN; \mathcal{I}) = D^{\mathcal{O}(\mathbf{WI}_{L-1}(\mathcal{I}))}$

(vertex prediction) $\quad \mathrm{sep}(GNN^{(t)}; \mathcal{I}) = D^{\mathcal{O}(\mathbf{WI}_{L-1,t}(\mathcal{I}))}$



Main Result: Strength of Interaction ∝ Walk Index

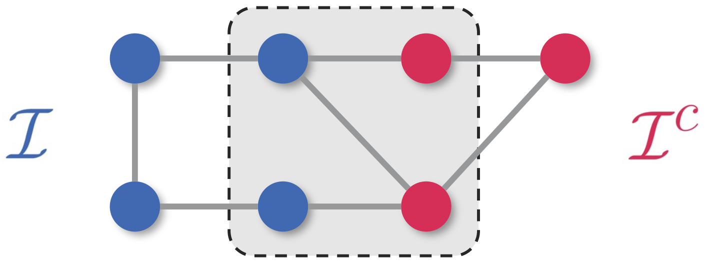
Theorem

For a depth L GNN with width D and $\mathcal{I} \subseteq V$:

(graph prediction) $$\mathrm{sep}(GNN; \mathcal{I}) = D^{\mathcal{O}(\mathbf{WI}_{L-1}(\mathcal{I}))}$$

(vertex prediction) $$\mathrm{sep}(GNN^{(t)}; \mathcal{I}) = D^{\mathcal{O}(\mathbf{WI}_{L-1,t}(\mathcal{I}))}$$

* Nearly matching lower bounds



Main Result: Strength of Interaction ∝ Walk Index

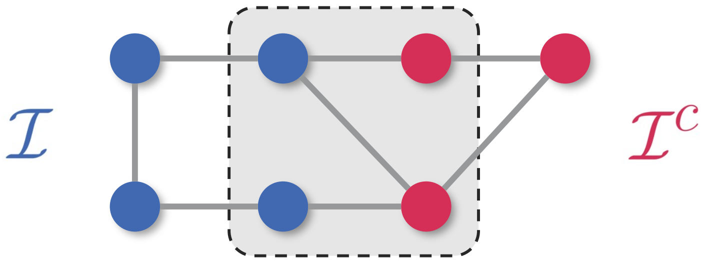
Theorem

For a depth L GNN with width D and $\mathcal{I} \subseteq V$:

(graph prediction) $$\mathrm{sep}(GNN; \mathcal{I}) = D^{\mathcal{O}(\mathbf{WI}_{L-1}(\mathcal{I}))}$$

(vertex prediction) $$\mathrm{sep}(GNN^{(t)}; \mathcal{I}) = D^{\mathcal{O}(\mathbf{WI}_{L-1,t}(\mathcal{I}))}$$

* Nearly matching lower bounds



ⓘ **Walk index of a partition controls strength of interaction**

Main Result: Strength of Interaction ∝ Walk Index

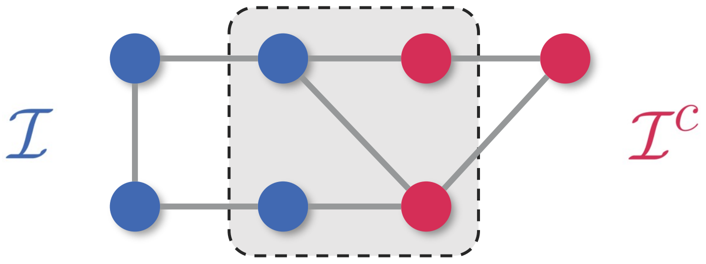
Theorem

For a depth L GNN with width D and $\mathcal{I} \subseteq V$:

(graph prediction) $$\mathrm{sep}(GNN; \mathcal{I}) = D^{\mathcal{O}(\mathbf{WI}_{L-1}(\mathcal{I}))}$$

(vertex prediction) $$\mathrm{sep}(GNN^{(t)}; \mathcal{I}) = D^{\mathcal{O}(\mathbf{WI}_{L-1,t}(\mathcal{I}))}$$

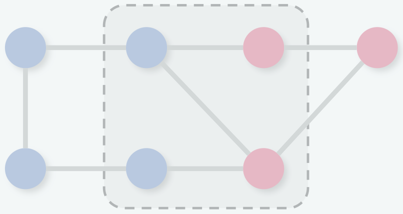
* Nearly matching lower bounds



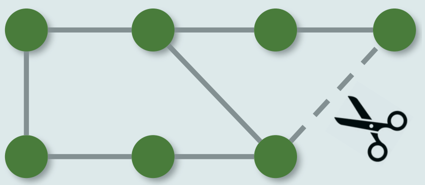
Walk index of a partition controls strength of interaction

Experiment: Implications of theory apply to widespread GNNs with **ReLU non-linearity** (GCN, GAT, GIN)

Main Contributions: Ability of GNNs to Model Interactions



Theory: Characterize ability of certain GNNs to model interactions between vertices



Application: Edge sparsification algorithm preserving interactions

Edge Sparsification

Edge Sparsification

Computations over large-scale graphs are **expensive**

Edge Sparsification

Computations over large-scale graphs are **expensive**

Edge Sparsification: Removing edges while maintaining graph properties

(e.g. Baswana & Sen 2007, Spielman & Srivastava 2011, Hamann et al 2016)

Edge Sparsification

Computations over large-scale graphs are **expensive**

Edge Sparsification: Removing edges while maintaining graph properties

(e.g. Baswana & Sen 2007, Spielman & Srivastava 2011, Hamann et al 2016)

In the context of GNNs, goal is to **maintain accuracy** when removing edges

Edge Sparsification

Computations over large-scale graphs are **expensive**

Edge Sparsification: Removing edges while maintaining graph properties

(e.g. Baswana & Sen 2007, Spielman & Srivastava 2011, Hamann et al 2016)

In the context of GNNs, goal is to **maintain accuracy** when removing edges

Our theory leads to a simple & effective algorithm for pruning edges

Algorithm: Walk Index Sparsification (WIS)

Algorithm: Walk Index Sparsification (WIS)

Idea: Greedily **prune edge whose removal harms interactions the least**

Algorithm: Walk Index Sparsification (WIS)

Idea: Greedily **prune edge whose removal harms interactions the least**

Theory: Leads to general scheme relying on **walk indices**

Algorithm: Walk Index Sparsification (WIS)

Idea: Greedily **prune edge whose removal harms interactions the least**

Theory: Leads to general scheme relying on **walk indices**

We focus here on vertex prediction (most relevant in large graphs)

Algorithm: Walk Index Sparsification (WIS)

Idea: Greedily **prune edge whose removal harms interactions the least**

Theory: Leads to general scheme relying on **walk indices**

We focus here on vertex prediction (most relevant in large graphs)

Algorithm: Until desired # edges are removed

Algorithm: Walk Index Sparsification (WIS)

Idea: Greedily **prune edge whose removal harms interactions the least**

Theory: Leads to general scheme relying on **walk indices**

We focus here on vertex prediction (most relevant in large graphs)

Algorithm: Until desired # edges are removed

(1) Per edge, imagine it removed and compute walk indices for preselected partitions

Algorithm: Walk Index Sparsification (WIS)

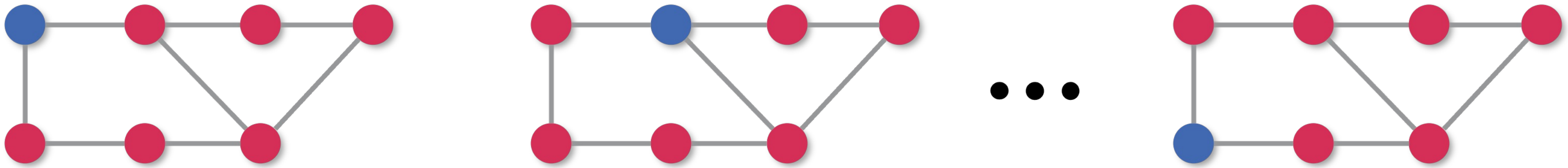
Idea: Greedily **prune edge whose removal harms interactions the least**

Theory: Leads to general scheme relying on **walk indices**

We focus here on vertex prediction (most relevant in large graphs)

Algorithm: Until desired # edges are removed

(1) Per edge, imagine it removed and compute walk indices for preselected partitions



Algorithm: Walk Index Sparsification (WIS)

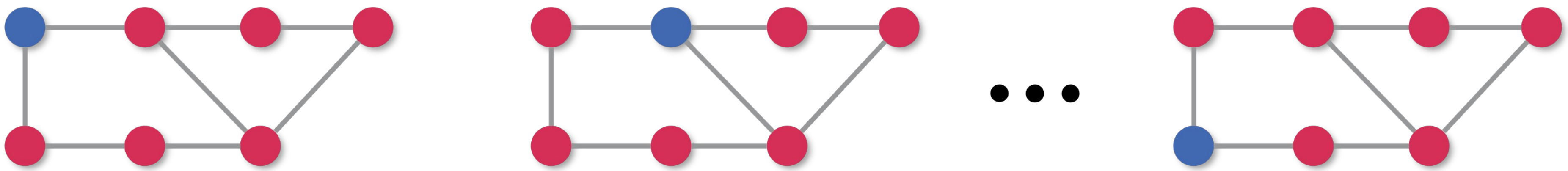
Idea: Greedily **prune edge whose removal harms interactions the least**

Theory: Leads to general scheme relying on **walk indices**

We focus here on vertex prediction (most relevant in large graphs)

Algorithm: Until desired # edges are removed

(1) Per edge, imagine it removed and compute walk indices for preselected partitions



(2) Remove edge that will keep maximal walk indices

Comparison of Edge Sparsification Methods

Experiment

Comparison of Edge Sparsification Methods

Experiment

Baselines: random, spectral (Spielman & Srivastava 2011), UGS (Chen et al. 2021)

Comparison of Edge Sparsification Methods

Experiment

Baselines: random, spectral (Spielman & Srivastava 2011), UGS (Chen et al. 2021)

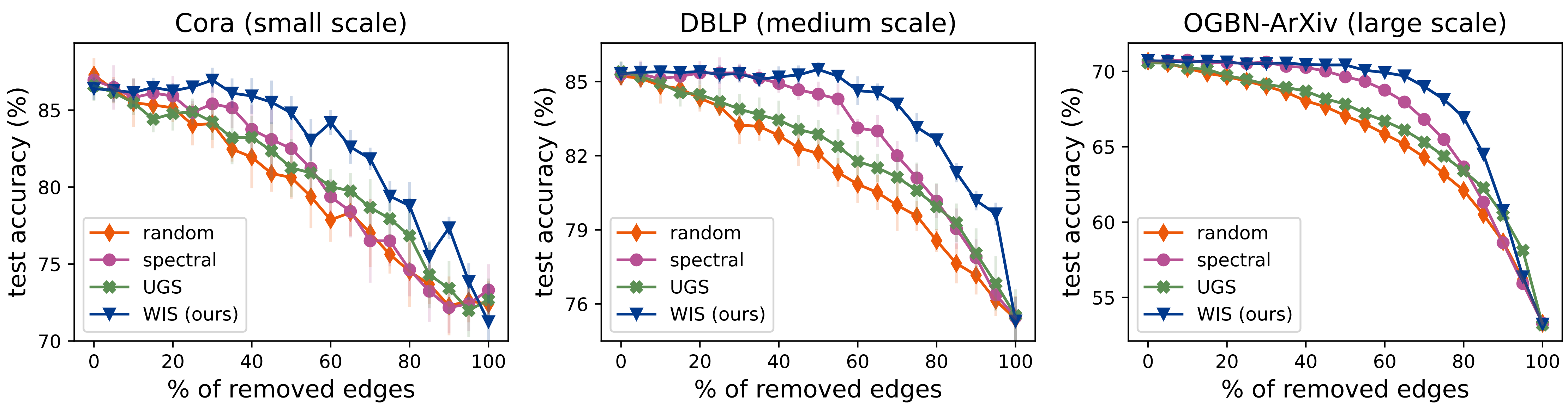
Model: depth $L = 3$ GCN (similar results using GIN & ResGCN and additional datasets)

Comparison of Edge Sparsification Methods

Experiment

Baselines: random, spectral (Spielman & Srivastava 2011), UGS (Chen et al. 2021)

Model: depth $L = 3$ GCN (similar results using GIN & ResGCN and additional datasets)

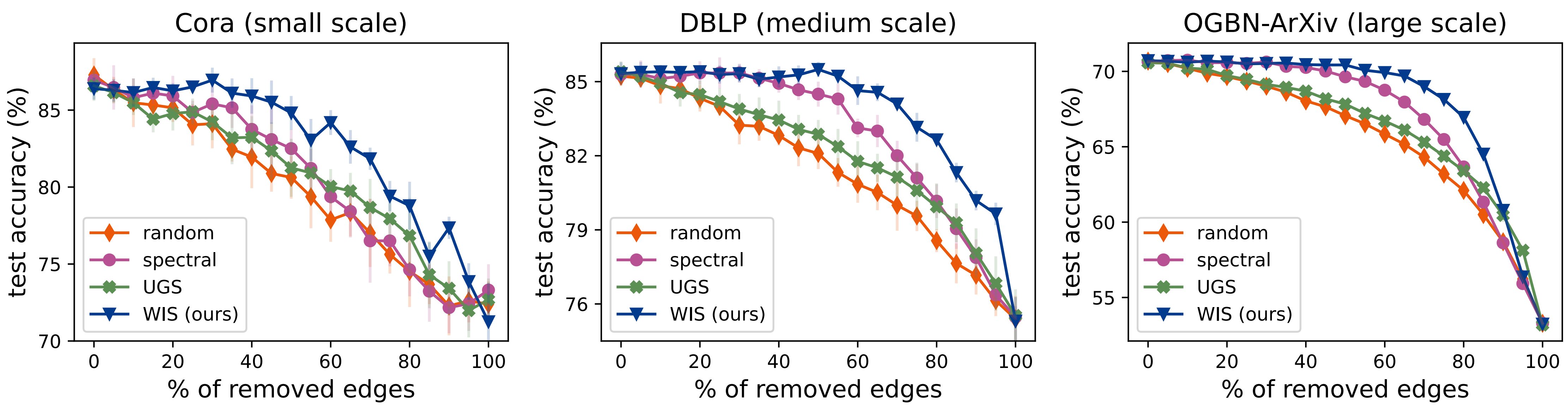


Comparison of Edge Sparsification Methods

Experiment

Baselines: random, spectral (Spielman & Srivastava 2011), UGS (Chen et al. 2021)

Model: depth $L = 3$ GCN (similar results using GIN & ResGCN and additional datasets)



ⓘ **WIS outperforms existing methods while being simple & efficient**

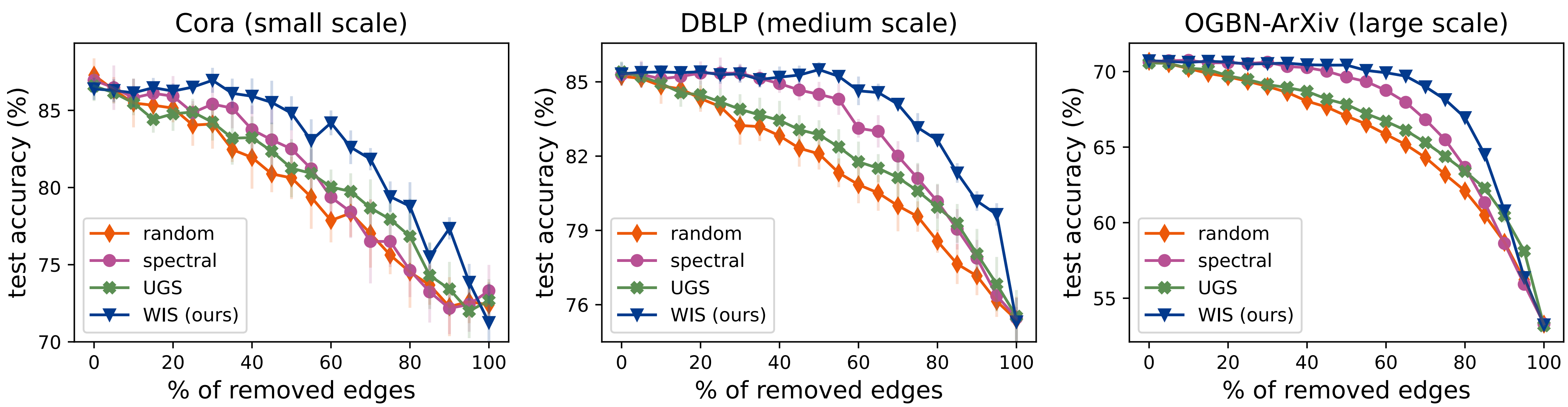
Comparison of Edge Sparsification Methods

Code

Experiment

Baselines: random, spectral (Spielman & Srivastava 2011), UGS (Chen et al. 2021)

Model: depth $L = 3$ GCN (similar results using GIN & ResGCN and additional datasets)



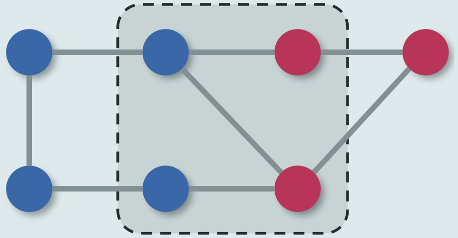
WIS outperforms existing methods while being simple & efficient

Conclusion: Ability of GNNs to Model Interactions

Conclusion: Ability of GNNs to Model Interactions

Theory

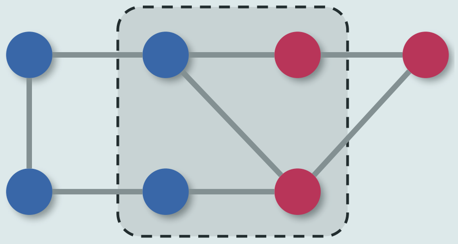
Walk index of a partition controls strength of interaction a GNN can model



Conclusion: Ability of GNNs to Model Interactions

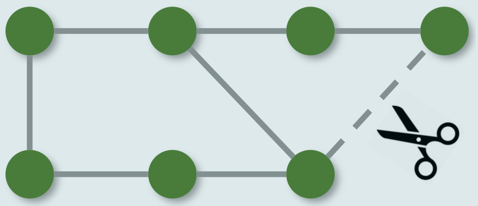
Theory

Walk index of a partition controls strength of interaction a GNN can model



Application

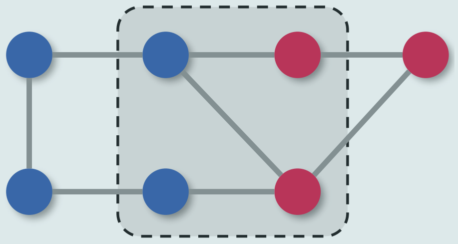
WIS: simple & efficient edge sparsification algorithm that outperforms alternative methods



Conclusion: Ability of GNNs to Model Interactions

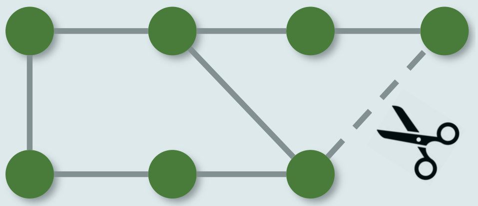
Theory

Walk index of a partition controls strength of interaction a GNN can model



Application

WIS: simple & efficient edge sparsification algorithm that outperforms alternative methods

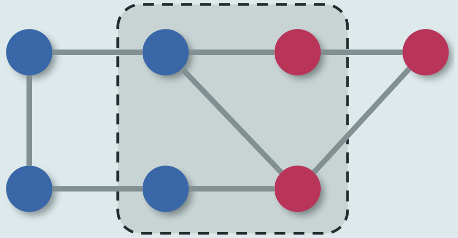


Going Forward: Studying modeled interactions may be key for

Conclusion: Ability of GNNs to Model Interactions

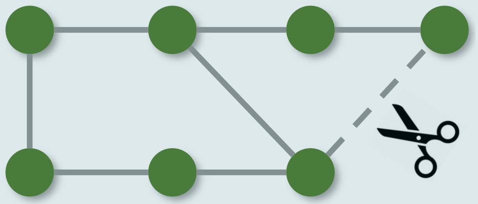
Theory

Walk index of a partition controls strength of interaction a GNN can model



Application

WIS: simple & efficient edge sparsification algorithm that outperforms alternative methods



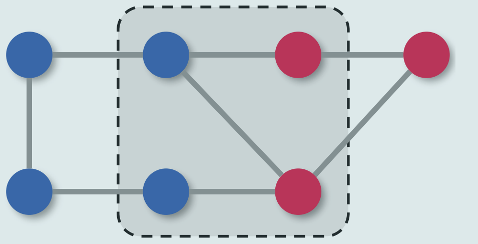
Going Forward: Studying modeled interactions may be key for

- Understanding aspects **beyond expressivity** (e.g. generalization)

Conclusion: Ability of GNNs to Model Interactions

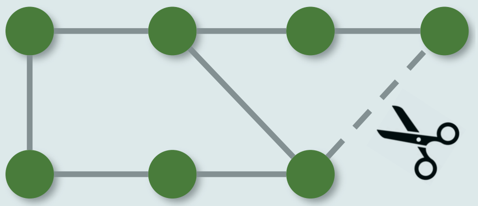
Theory

Walk index of a partition controls strength of interaction a GNN can model



Application

WIS: simple & efficient edge sparsification algorithm that outperforms alternative methods



Going Forward: Studying modeled interactions may be key for

- Understanding aspects beyond expressivity (e.g. generalization)
- Improving performance of GNNs beyond edge sparsification

Vanishing Gradients in Reinforcement Finetuning of Language Models

Language Models (LMs)

Language Models (LMs)

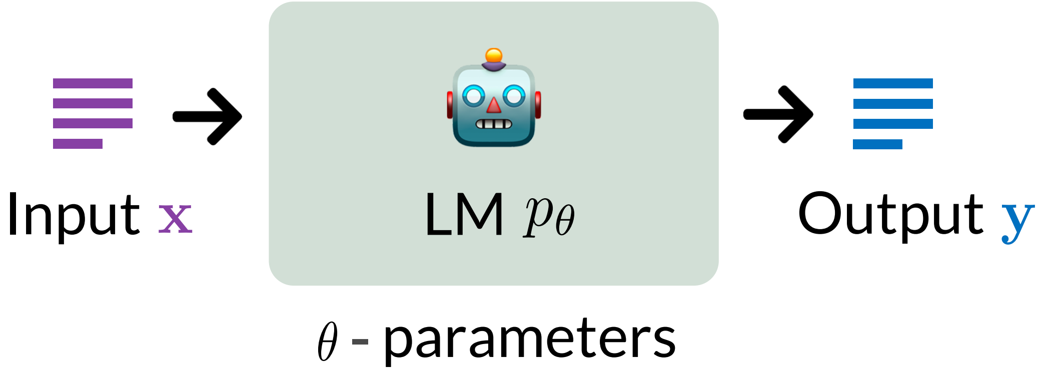
Language Model (LM): Neural network trained on large amounts of (internet) text data to produce a **distribution over text**

LM p_θ

θ - parameters

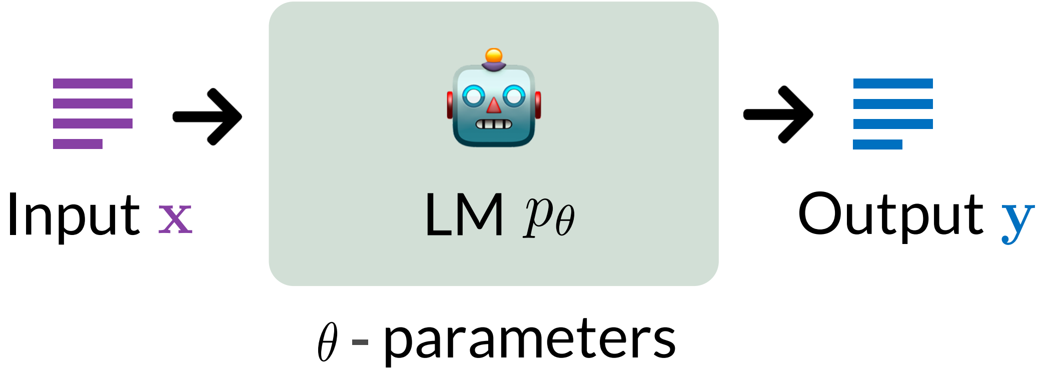
Language Models (LMs)

Language Model (LM): Neural network trained on large amounts of (internet) text data to produce a **distribution over text**



Language Models (LMs)

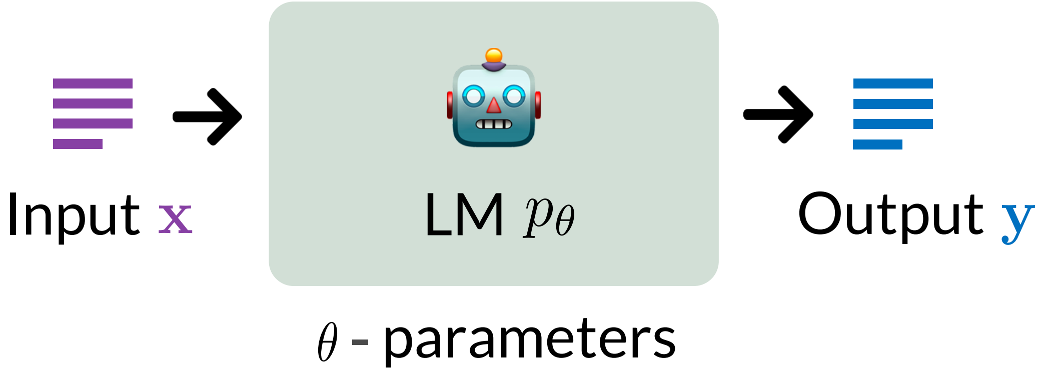
Language Model (LM): Neural network trained on large amounts of (internet) text data to produce a **distribution over text**



LMs are typically autoregressive

Language Models (LMs)

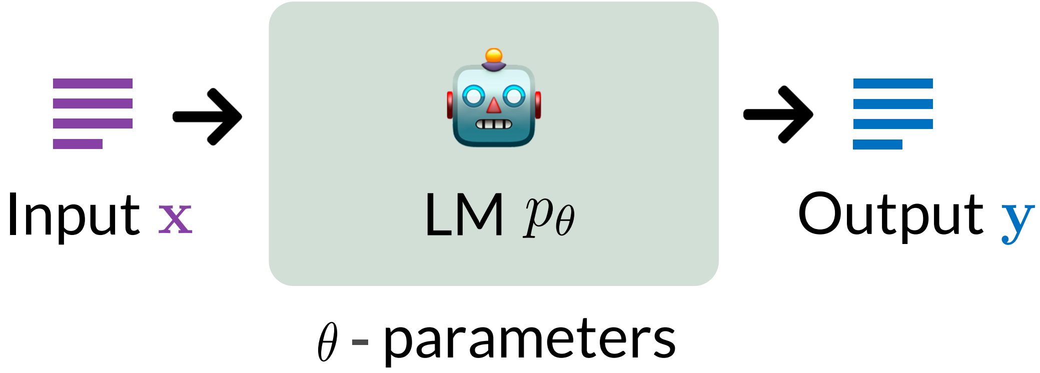
Language Model (LM): Neural network trained on large amounts of (internet) text data to produce a **distribution over text**



LMs are typically autoregressive $p_\theta(\mathbf{y}|\mathbf{x}) = \prod_{l=1}^{L} p_\theta(y_l|\mathbf{x}, \mathbf{y}_{\leq l-1})$

Language Models (LMs)

Language Model (LM): Neural network trained on large amounts of (internet) text data to produce a **distribution over text**



LMs are typically autoregressive $p_\theta(\mathbf{y}|\mathbf{x}) = \prod_{l=1}^{L} \underbrace{p_\theta(y_l|\mathbf{x}, \mathbf{y}_{\leq l-1})}$

softmax is used for producing next-token probabilities

Supervised Finetuning of LMs

LMs are adapted to human preferences and downstream tasks via **finetuning**

Supervised Finetuning of LMs

LMs are adapted to human preferences and downstream tasks via **finetuning**

Supervised Finetuning (SFT)

Minimize cross entropy loss over labeled inputs via **gradient-based methods**

Supervised Finetuning of LMs

LMs are adapted to human preferences and downstream tasks via **finetuning**

Supervised Finetuning (SFT)

Minimize cross entropy loss over labeled inputs via **gradient-based methods**

Limitations:

Supervised Finetuning of LMs

LMs are adapted to human preferences and downstream tasks via **finetuning**

Supervised Finetuning (SFT)

Minimize cross entropy loss over labeled inputs via **gradient-based methods**

Limitations:

Hard to formalize human preferences through labels

Supervised Finetuning of LMs

LMs are adapted to human preferences and downstream tasks via **finetuning**

Supervised Finetuning (SFT)

Minimize cross entropy loss over labeled inputs via **gradient-based methods**

Limitations:

- Hard to formalize human preferences through labels
- Labeled data is expensive

Reinforcement Finetuning of LMs

Limitations of SFT led to wide adoption of a **reinforcement learning**-based approach

(e.g. Ziegler et al. 2019, Stiennon et al. 2020, Ouyang et al. 2022, Bai et al. 2022, Dubois et al. 2023, Touvron et al. 2023)

Reinforcement Finetuning of LMs

Limitations of SFT led to wide adoption of a **reinforcement learning**-based approach

(e.g. Ziegler et al. 2019, Stiennon et al. 2020, Ouyang et al. 2022, Bai et al. 2022, Dubois et al. 2023, Touvron et al. 2023)

Reinforcement Finetuning (RFT)

Maximize reward over unlabeled inputs via **policy gradient algorithms**

reward function $r(\mathbf{x}, \mathbf{y})$

Reinforcement Finetuning of LMs

Limitations of SFT led to wide adoption of a **reinforcement learning**-based approach

(e.g. Ziegler et al. 2019, Stiennon et al. 2020, Ouyang et al. 2022, Bai et al. 2022, Dubois et al. 2023, Touvron et al. 2023)

Reinforcement Finetuning (RFT)

Maximize reward over unlabeled inputs via **policy gradient algorithms**

reward function $r(\mathbf{x}, \mathbf{y})$

Expected reward for input $\mathbf{x}$: $V_\theta(\mathbf{x}) = \mathbb{E}_{\mathbf{y} \sim p_\theta(\cdot|\mathbf{x})} [r(\mathbf{x}, \mathbf{y})]$

Reinforcement Finetuning of LMs

Limitations of SFT led to wide adoption of a **reinforcement learning**-based approach
(e.g. Ziegler et al. 2019, Stiennon et al. 2020, Ouyang et al. 2022, Bai et al. 2022, Dubois et al. 2023, Touvron et al. 2023)

Reinforcement Finetuning (RFT)

Maximize reward over unlabeled inputs via **policy gradient algorithms**

reward function $r(\mathbf{x}, \mathbf{y})$

Expected reward for input $\mathbf{x}$: $V_\theta(\mathbf{x}) = \mathbb{E}_{\mathbf{y} \sim p_\theta(\cdot|\mathbf{x})} [r(\mathbf{x}, \mathbf{y})]$

Reward function $r(\mathbf{x}, \mathbf{y})$ can be:

Reinforcement Finetuning of LMs

Limitations of SFT led to wide adoption of a **reinforcement learning**-based approach

(e.g. Ziegler et al. 2019, Stiennon et al. 2020, Ouyang et al. 2022, Bai et al. 2022, Dubois et al. 2023, Touvron et al. 2023)

Reinforcement Finetuning (RFT)

Maximize reward over unlabeled inputs via **policy gradient algorithms**

, , $\cdots$, reward function $r(\mathbf{x}, \mathbf{y})$

Expected reward for input $\mathbf{x}$: $V_\theta(\mathbf{x}) = \mathbb{E}_{\mathbf{y} \sim p_\theta(\cdot|\mathbf{x})} [r(\mathbf{x}, \mathbf{y})]$

Reward function $r(\mathbf{x}, \mathbf{y})$ can be:

- Learned from human preferences

Reinforcement Finetuning of LMs

Limitations of SFT led to wide adoption of a **reinforcement learning**-based approach
(e.g. Ziegler et al. 2019, Stiennon et al. 2020, Ouyang et al. 2022, Bai et al. 2022, Dubois et al. 2023, Touvron et al. 2023)

Reinforcement Finetuning (RFT)

Maximize reward over unlabeled inputs via **policy gradient algorithms**

reward function $r(\mathbf{x}, \mathbf{y})$

Expected reward for input $\mathbf{x}$: $V_\theta(\mathbf{x}) = \mathbb{E}_{\mathbf{y} \sim p_\theta(\cdot|\mathbf{x})} [r(\mathbf{x}, \mathbf{y})]$

Reward function $r(\mathbf{x}, \mathbf{y})$ can be:

- Learned from human preferences
- Tailored to a downstream task

Main Contributions: Vanishing Gradients in RFT

Main Contributions: Vanishing Gradients in RFT

$\nabla_\theta \mathbf{V}_\theta(\mathbf{x}) \approx \mathbf{0}$ Fundamental vanishing gradients problem in RFT

Main Contributions: Vanishing Gradients in RFT

$\nabla_\theta \mathbf{V}_\theta(\mathbf{x}) \approx \mathbf{0}$ Fundamental vanishing gradients problem in RFT

Vanishing gradients are prevalent and harm ability to maximize reward

Main Contributions: Vanishing Gradients in RFT

$\nabla_\theta \mathbf{V}_\theta(\mathbf{x}) \approx \mathbf{0}$ Fundamental vanishing gradients problem in RFT

Vanishing gradients are prevalent and harm ability to maximize reward

Exploring ways to overcome vanishing gradients in RFT

Main Contributions: Vanishing Gradients in RFT

$\nabla_\theta \mathbf{V}_\theta(\mathbf{x}) \approx \mathbf{0}$ Fundamental vanishing gradients problem in RFT

Vanishing gradients are prevalent and harm ability to maximize reward

Exploring ways to overcome vanishing gradients in RFT

Vanishing Gradients Due to Small Reward Standard Deviation (STD)

$\mathrm{STD}_{\mathbf{y} \sim p_\theta(\cdot|\mathbf{x})}[r(\mathbf{x}, \mathbf{y})]$ — reward std of $\mathbf{x}$ under the model

Vanishing Gradients Due to Small Reward Standard Deviation (STD)

$\mathrm{STD}_{\mathbf{y}\sim p_\theta(\cdot|\mathbf{x})}[r(\mathbf{x},\mathbf{y})]$ — reward std of $\mathbf{x}$ under the model

Theorem

$$\|\nabla_\theta V_\theta(\mathbf{x})\| = O\big(\mathrm{STD}_{\mathbf{y}\sim p_\theta(\cdot|\mathbf{x})}[r(\mathbf{x},\mathbf{y})]^{2/3}\big)$$

*Same holds for PPO gradient

Vanishing Gradients Due to Small Reward Standard Deviation (STD)

$\mathrm{STD}_{\mathbf{y} \sim p_\theta(\cdot|\mathbf{x})}[r(\mathbf{x}, \mathbf{y})]$ — reward std of $\mathbf{x}$ under the model

Theorem

$$\|\nabla_\theta V_\theta(\mathbf{x})\| = O\left(\mathrm{STD}_{\mathbf{y} \sim p_\theta(\cdot|\mathbf{x})}[r(\mathbf{x}, \mathbf{y})]^{2/3}\right)$$

Expected gradient for an input vanishes when reward std is small, even if reward mean is suboptimal

*Same holds for PPO gradient

Vanishing Gradients Due to Small Reward Standard Deviation (STD)

$\mathrm{STD}_{\mathbf{y}\sim p_\theta(\cdot|\mathbf{x})}[r(\mathbf{x},\mathbf{y})]$ — reward std of $\mathbf{x}$ under the model

Theorem

$$\|\nabla_\theta V_\theta(\mathbf{x})\| = O\left(\mathrm{STD}_{\mathbf{y}\sim p_\theta(\cdot|\mathbf{x})}[r(\mathbf{x},\mathbf{y})]^{2/3}\right)$$

Expected gradient for an input vanishes when reward std is small, even if reward mean is suboptimal

*Same holds for PPO gradient

Proof Idea: Stems from use of softmax + reward maximization objective

Vanishing Gradients Due to Small Reward Standard Deviation (STD)

$\mathrm{STD}_{\mathbf{y}\sim p_\theta(\cdot|\mathbf{x})}[r(\mathbf{x},\mathbf{y})]$ — reward std of $\mathbf{x}$ under the model

Theorem

$$\|\nabla_\theta V_\theta(\mathbf{x})\| = O\left(\mathrm{STD}_{\mathbf{y}\sim p_\theta(\cdot|\mathbf{x})}[r(\mathbf{x},\mathbf{y})]^{2/3}\right)$$

Expected gradient for an input vanishes when reward std is small, even if reward mean is suboptimal

*Same holds for PPO gradient

Proof Idea: Stems from use of softmax + reward maximization objective

Note: Bound applies to expected gradients of individual inputs (as opposed to of batch/population)

Vanishing Gradients Due to Small Reward Standard Deviation (STD)

$\mathrm{STD}_{\mathbf{y} \sim p_\theta(\cdot|\mathbf{x})}[r(\mathbf{x}, \mathbf{y})]$ — reward std of $\mathbf{x}$ under the model

Theorem

$$\|\nabla_\theta V_\theta(\mathbf{x})\| = O\left(\mathrm{STD}_{\mathbf{y} \sim p_\theta(\cdot|\mathbf{x})}[r(\mathbf{x}, \mathbf{y})]^{2/3}\right)$$

Expected gradient for an input vanishes when reward std is small, even if reward mean is suboptimal

*Same holds for PPO gradient

Proof Idea: Stems from use of softmax + reward maximization objective

Note: Bound applies to expected gradients of individual inputs (as opposed to of batch/population)

Can be problematic when finetuning text distribution differs from pretraining

Main Contributions: Vanishing Gradients in RFT

$\nabla_\theta \mathbf{V}_\theta(\mathbf{x}) \approx \mathbf{0}$ Fundamental vanishing gradients problem in RFT

Vanishing gradients are prevalent and harm ability to maximize reward

Exploring ways to overcome vanishing gradients in RFT

Prevalence and Detrimental Effects of Vanishing Gradients

Prevalence and Detrimental Effects of Vanishing Gradients

Benchmark: GRUE (Ramamurthy et al. 2023)
7 language generation datasets

Prevalence and Detrimental Effects of Vanishing Gradients

Benchmark: GRUE (Ramamurthy et al. 2023)
7 language generation datasets

Models: GPT-2 and T5-base

Prevalence and Detrimental Effects of Vanishing Gradients

Benchmark: GRUE (Ramamurthy et al. 2023)
7 language generation datasets

Models: GPT-2 and T5-base

Finding I

3 of 7 datasets contain considerable # of train inputs with small reward std and low reward

Prevalence and Detrimental Effects of Vanishing Gradients

Benchmark: GRUE (Ramamurthy et al. 2023)
7 language generation datasets

Models: GPT-2 and T5-base

vanishing gradients

Finding I

3 of 7 datasets contain considerable # of train inputs with small reward std and low reward

Prevalence and Detrimental Effects of Vanishing Gradients

Benchmark: GRUE (Ramamurthy et al. 2023)
7 language generation datasets

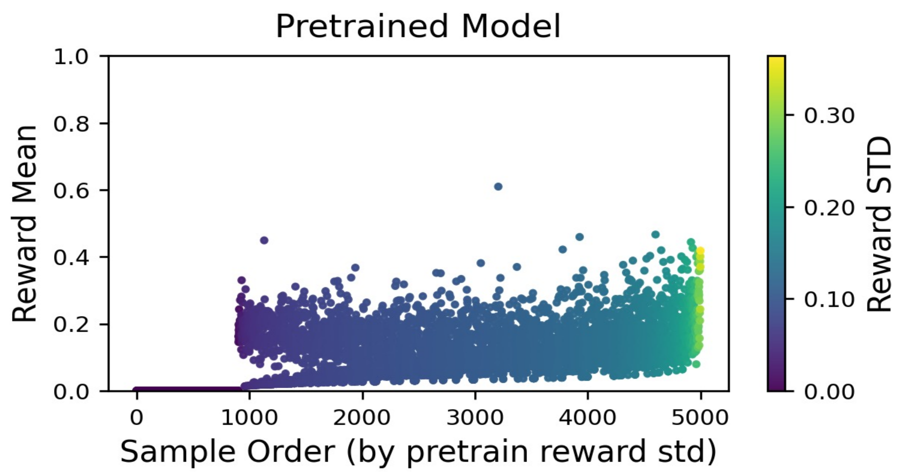
Models: GPT-2 and T5-base

vanishing gradients

Finding I

3 of 7 datasets contain considerable # of train inputs with small reward std and low reward

NarrativeQA
(many inputs w/ small std)



Prevalence and Detrimental Effects of Vanishing Gradients

Benchmark: GRUE (Ramamurthy et al. 2023)
7 language generation datasets

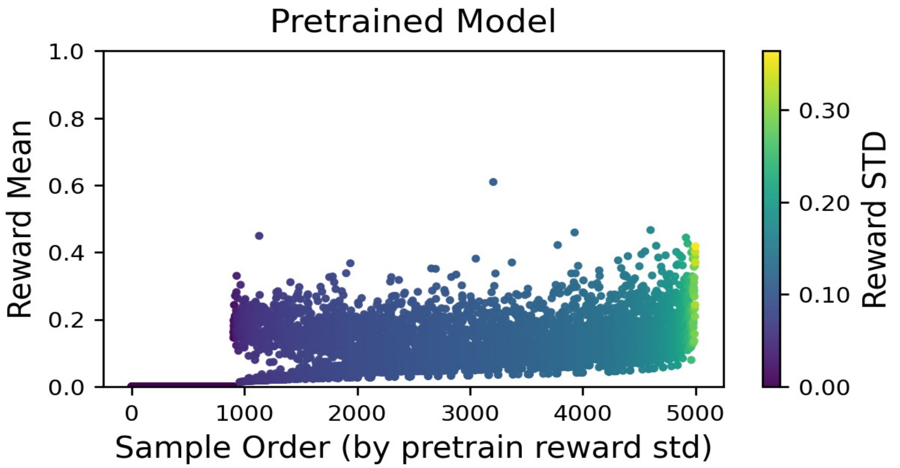
Models: GPT-2 and T5-base

vanishing gradients

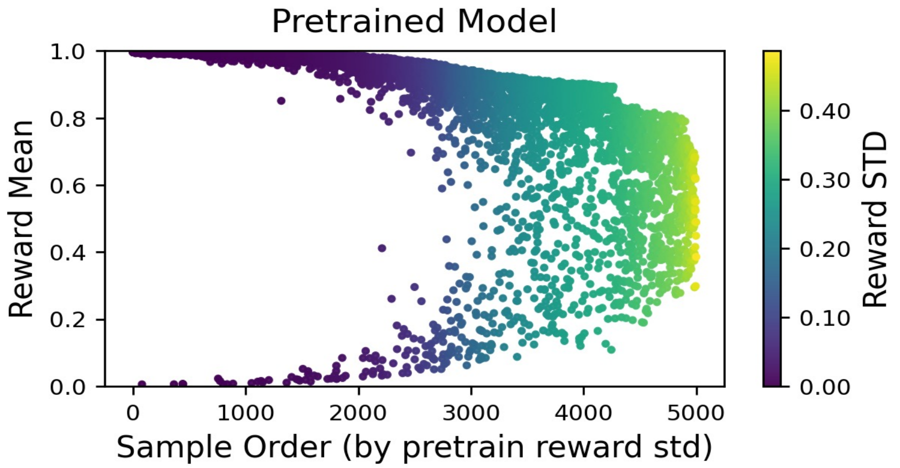
Finding I

3 of 7 datasets contain considerable # of train inputs with small reward std and low reward

NarrativeQA
(many inputs w/ small std)



IMDB
(few inputs w/ small std)



Prevalence and Detrimental Effects of Vanishing Gradients

Benchmark: GRUE (Ramamurthy et al. 2023)
7 language generation datasets

Models: GPT-2 and T5-base

Prevalence and Detrimental Effects of Vanishing Gradients

Benchmark: GRUE (Ramamurthy et al. 2023)
7 language generation datasets

Models: GPT-2 and T5-base

Finding II

As expected, RFT has limited impact on the reward of inputs with small reward std

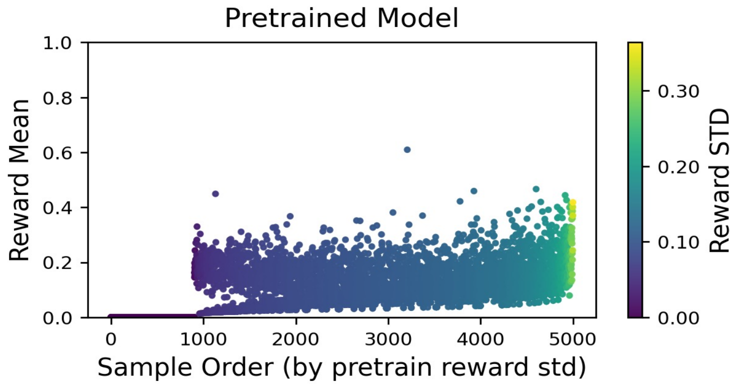
Prevalence and Detrimental Effects of Vanishing Gradients

Benchmark: GRUE (Ramamurthy et al. 2023) Models: GPT-2 and T5-base
7 language generation datasets

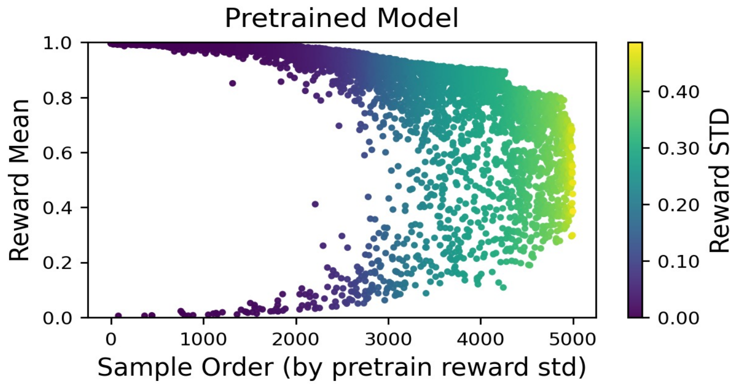
Finding II

As expected, RFT has limited impact on the reward of inputs with small reward std

NarrativeQA
(many inputs w/ small std)



IMDB
(few inputs w/ small std)



Prevalence and Detrimental Effects of Vanishing Gradients

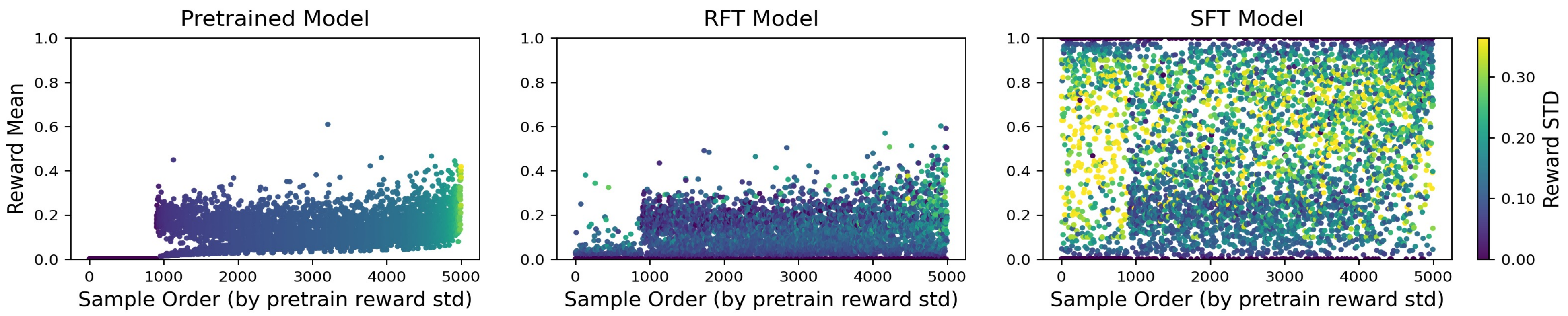
Benchmark: GRUE (Ramamurthy et al. 2023)
7 language generation datasets

Models: GPT-2 and T5-base

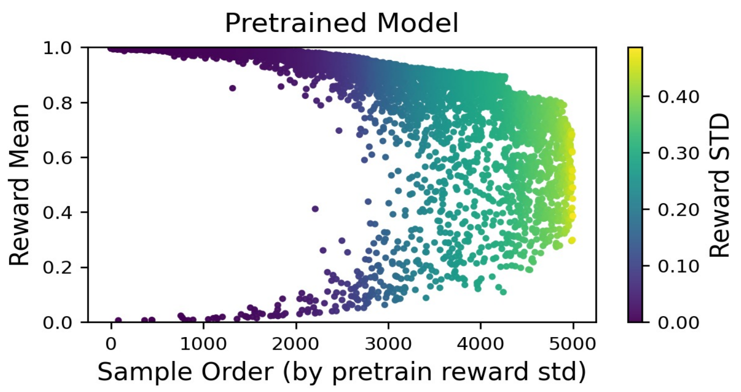
Finding II

As expected, RFT has limited impact on the reward of inputs with small reward std

NarrativeQA
(many inputs w/ small std)



IMDB
(few inputs w/ small std)



Prevalence and Detrimental Effects of Vanishing Gradients

Benchmark: GRUE (Ramamurthy et al. 2023)
7 language generation datasets

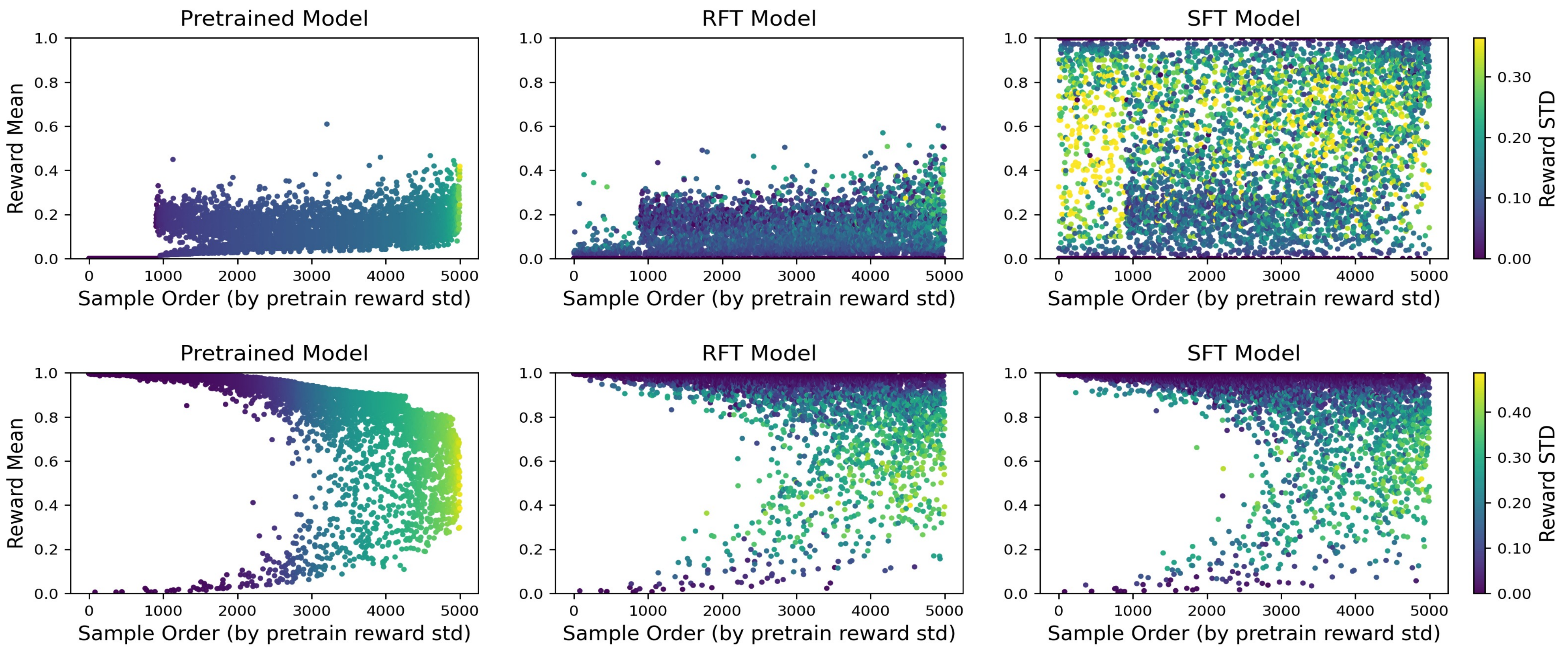
Models: GPT-2 and T5-base

Finding II

As expected, RFT has limited impact on the reward of inputs with small reward std

NarrativeQA
(many inputs w/ small std)

IMDB
(few inputs w/ small std)



Prevalence and Detrimental Effects of Vanishing Gradients

Benchmark: GRUE (Ramamurthy et al. 2023)
7 language generation datasets

Models: GPT-2 and T5-base

Prevalence and Detrimental Effects of Vanishing Gradients

Benchmark: GRUE (Ramamurthy et al. 2023)
7 language generation datasets

Models: GPT-2 and T5-base

Finding III

RFT performance is worse when inputs with small reward std are prevalent

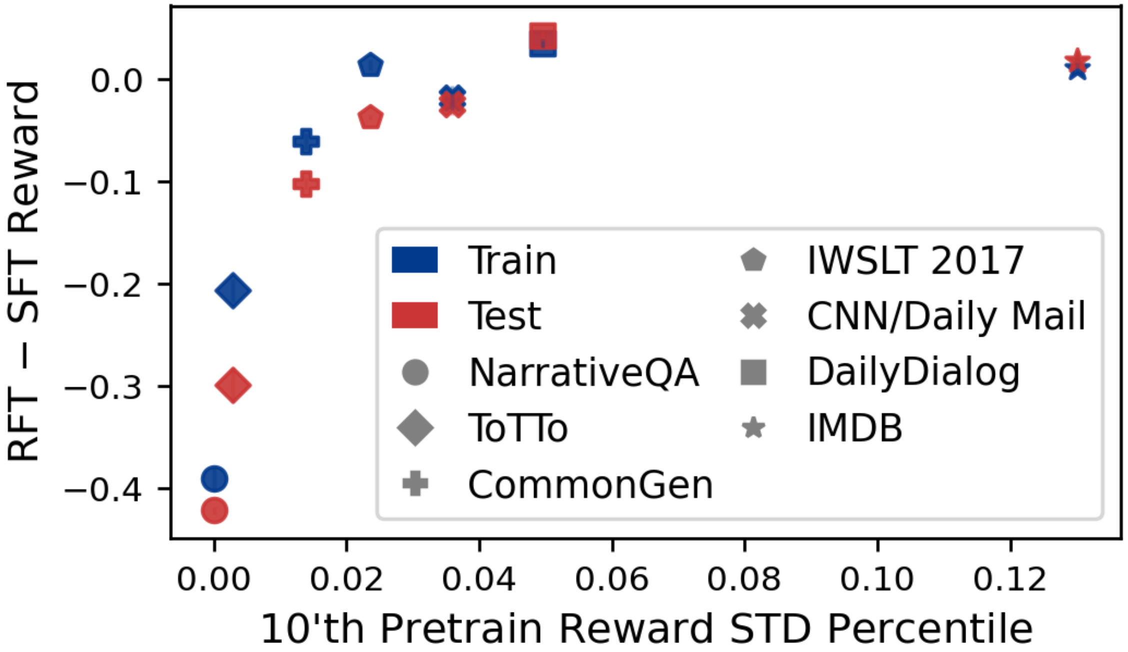
Prevalence and Detrimental Effects of Vanishing Gradients

Benchmark: GRUE (Ramamurthy et al. 2023)
7 language generation datasets

Models: GPT-2 and T5-base

Finding III

RFT performance is worse when inputs with small reward std are prevalent



Vanishing Gradients or Insufficient Exploration?

We saw that **vanishing expected gradients is indicative of RFT performance**

Vanishing Gradients or Insufficient Exploration?

We saw that **vanishing expected gradients is indicative of RFT performance**

(vanishing expected gradients: measured by reward std)

Vanishing Gradients or Insufficient Exploration?

We saw that **vanishing expected gradients is indicative of RFT performance**

measured by reward std

Possible Confounding Factor: Insufficient Exploration

Large output space in language generation → challenge of exploration

(e.g. Ranzato et al. 2016, Choshen et al. 2020)

Vanishing Gradients or Insufficient Exploration?

We saw that vanishing expected gradients is indicative of RFT performance

measured by reward std

Possible Confounding Factor: Insufficient Exploration

Large output space in language generation → challenge of exploration
(e.g. Ranzato et al. 2016, Choshen et al. 2020)

→ challenge of accurately estimating $\nabla_\theta V_\theta(\mathbf{x})$

Vanishing Gradients or Insufficient Exploration?

We saw that **vanishing expected gradients is indicative of RFT performance**

measured by reward std

Possible Confounding Factor: Insufficient Exploration

Large output space in language generation → challenge of exploration
(e.g. Ranzato et al. 2016, Choshen et al. 2020)

→ challenge of accurately estimating $\nabla_\theta V_\theta(\mathbf{x})$

Q: Does the difficulty of RFT to maximize reward stem from vanishing gradients or just insufficient exploration?

Vanishing Gradients or Insufficient Exploration?

We saw that vanishing expected gradients is indicative of RFT performance

measured by reward std

Possible Confounding Factor: Insufficient Exploration

Large output space in language generation → challenge of exploration
(e.g. Ranzato et al. 2016, Choshen et al. 2020)

→ challenge of accurately estimating $\nabla_\theta V_\theta(\mathbf{x})$

Q: Does the difficulty of RFT to maximize reward stem from vanishing gradients or just insufficient exploration?

We address Q via controlled experiments and theoretical analysis

Controlled Experiments and Theoretical Analysis

Controlled Experiments and Theoretical Analysis

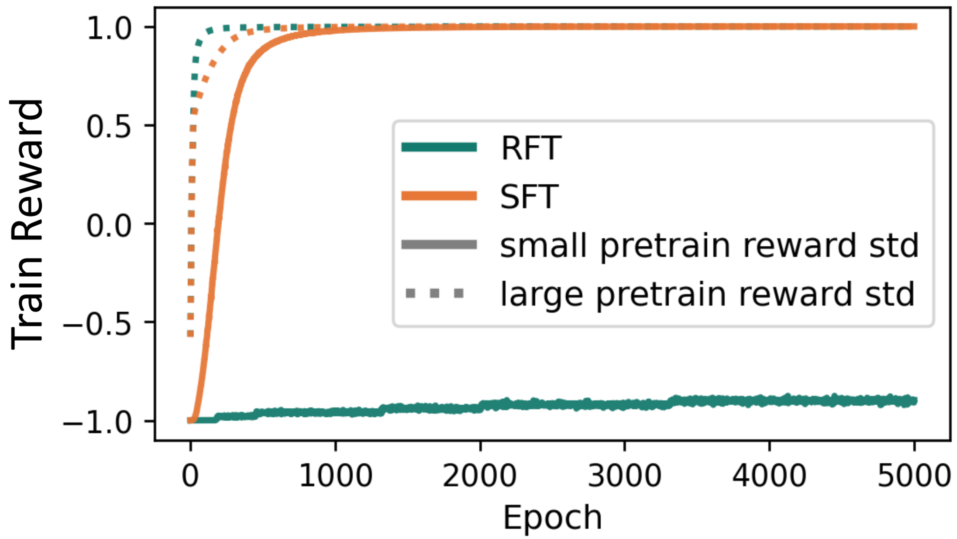
Controlled Experiments

Environments with **perfect exploration**,
i.e. RFT has access to expected gradients

Controlled Experiments and Theoretical Analysis

Controlled Experiments

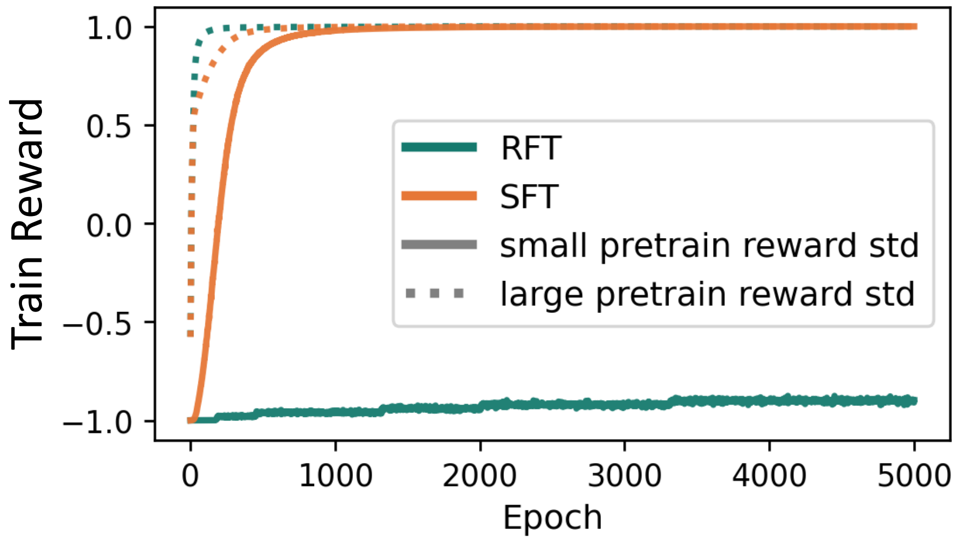
Environments with **perfect exploration**,
i.e. RFT has access to expected gradients



Controlled Experiments and Theoretical Analysis

Controlled Experiments

Environments with **perfect exploration**, i.e. RFT has access to expected gradients



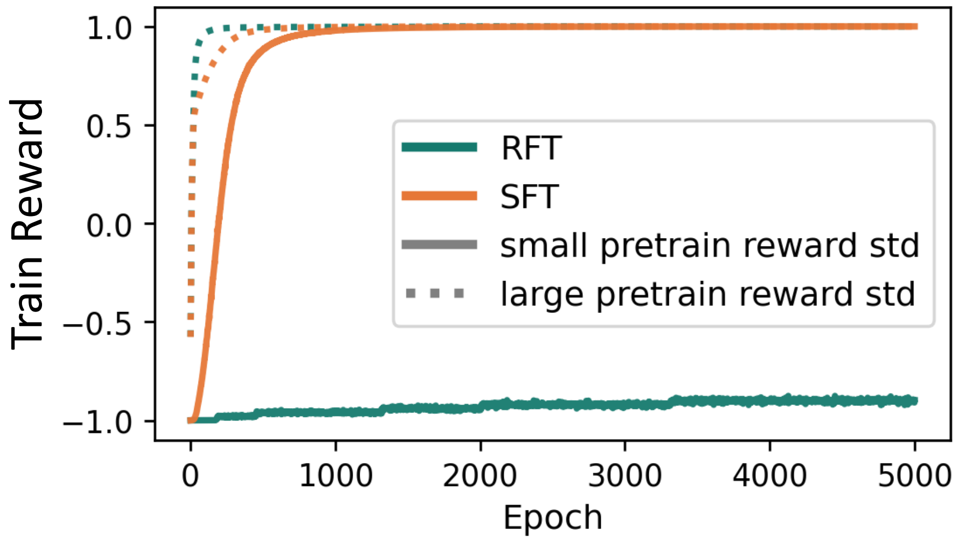
Theoretical Analysis

Simplified setting of linear classification over orthonormal data with **perfect exploration**

Controlled Experiments and Theoretical Analysis

Controlled Experiments

Environments with **perfect exploration**, i.e. RFT has access to expected gradients



Theoretical Analysis

Simplified setting of linear classification over orthonormal data with **perfect exploration**

Theorem

Time it takes to correctly classify input $\mathbf{x}$ is:

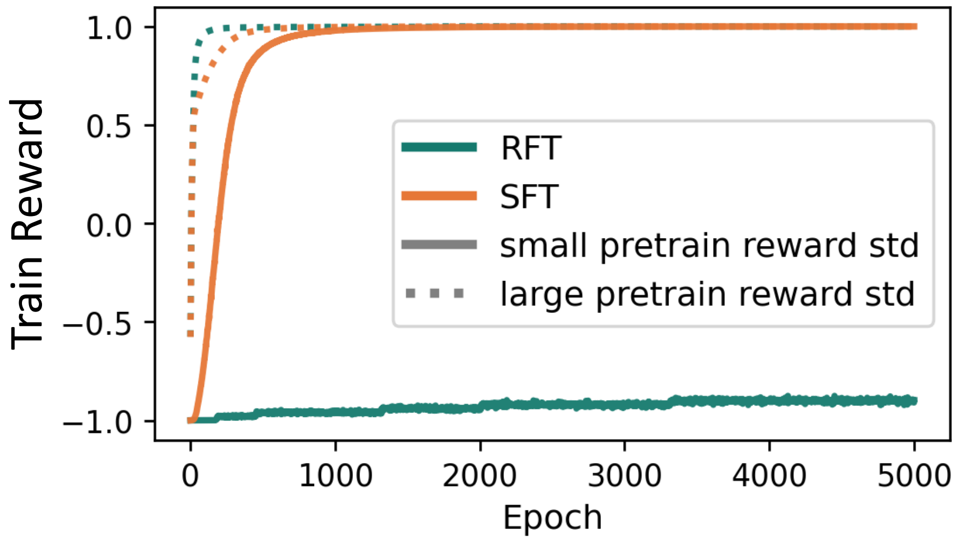
in RFT - $\Omega\left(1/\mathrm{STD}_{\mathbf{y}\sim p_{\theta(0)}(\cdot|\mathbf{x})}[r(\mathbf{x},\mathbf{y})]^2\right)$

in SFT - $O\left(\ln\left(1/\mathrm{STD}_{\mathbf{y}\sim p_{\theta(0)}(\cdot|\mathbf{x})}[r(\mathbf{x},\mathbf{y})]\right)\right)$

Controlled Experiments and Theoretical Analysis

Controlled Experiments

Environments with **perfect exploration**, i.e. RFT has access to expected gradients



Theoretical Analysis

Simplified setting of linear classification over orthonormal data with **perfect exploration**

Theorem

Time it takes to correctly classify input $\mathbf{x}$ is:

in RFT - $\Omega\left(1/\mathrm{STD}_{\mathbf{y}\sim p_{\theta(0)}(\cdot|\mathbf{x})}[r(\mathbf{x},\mathbf{y})]^2\right)$

in SFT - $O\left(\ln\left(1/\mathrm{STD}_{\mathbf{y}\sim p_{\theta(0)}(\cdot|\mathbf{x})}[r(\mathbf{x},\mathbf{y})]\right)\right)$

ⓘ RFT struggles to maximize reward for inputs with small reward std despite perfect exploration

Main Contributions: Vanishing Gradients in RFT

$\nabla_\theta \mathbf{V}_\theta(\mathbf{x}) \approx \mathbf{0}$ Fundamental vanishing gradients problem in RFT

Vanishing gradients are prevalent and harm ability to maximize reward

Exploring ways to overcome vanishing gradients in RFT

Overcoming Vanishing Gradients in RFT

Overcoming Vanishing Gradients in RFT

Common Heuristics: Increasing learning rate, temperature, entropy regularization

Overcoming Vanishing Gradients in RFT

Common Heuristics: Increasing learning rate, temperature, entropy regularization ❌

Overcoming Vanishing Gradients in RFT

Common Heuristics: Increasing learning rate, temperature, entropy regularization ❌

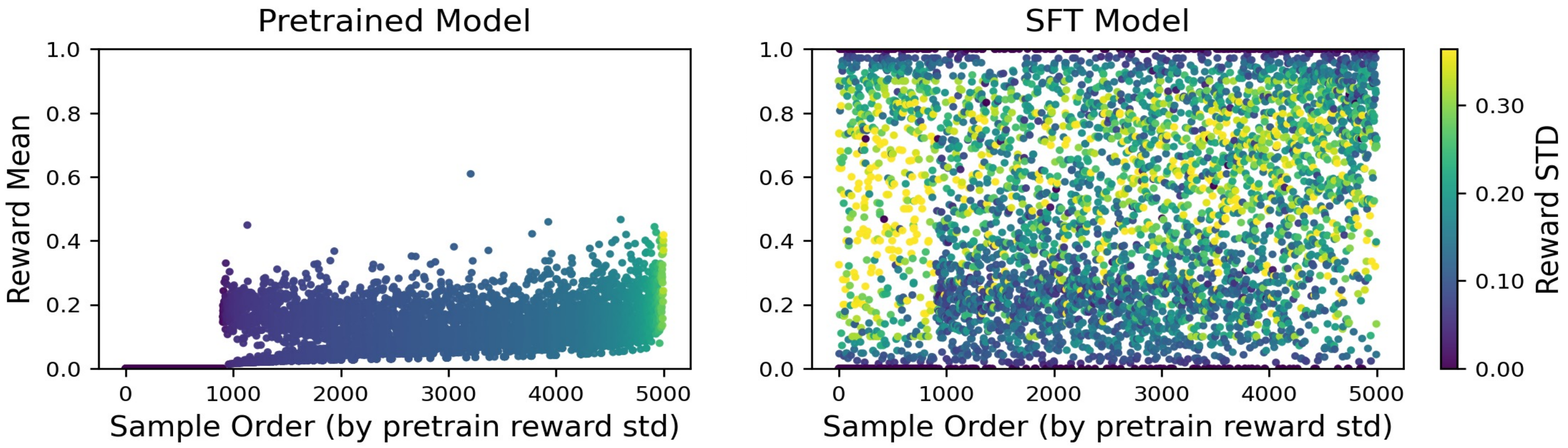
Observation: Initial SFT phase reduces number of inputs with small reward std

Overcoming Vanishing Gradients in RFT

Common Heuristics: Increasing learning rate, temperature, entropy regularization ❌

Observation: Initial SFT phase reduces number of inputs with small reward std

NarrativeQA (train)

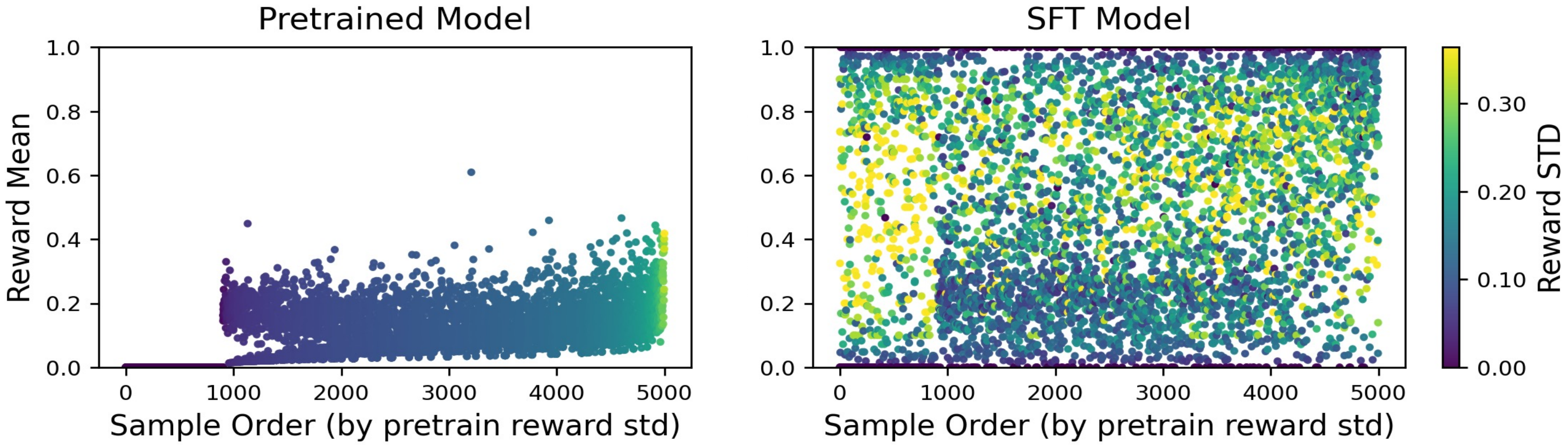


Overcoming Vanishing Gradients in RFT

Common Heuristics: Increasing learning rate, temperature, entropy regularization ❌

Observation: Initial SFT phase reduces number of inputs with small reward std

NarrativeQA (train)



ⓘ **Importance of SFT in RFT pipeline: mitigates vanishing gradients**

A Few SFT Steps on a Small Number of Samples Suffice

A Few SFT Steps on a Small Number of Samples Suffice

Limitation of Initial SFT Phase: Requires labeled data

A Few SFT Steps on a Small Number of Samples Suffice

Limitation of Initial SFT Phase: Requires labeled data

Expectation: If SFT phase is beneficial due to mitigating vanishing gradients for RFT

A Few SFT Steps on a Small Number of Samples Suffice

Limitation of Initial SFT Phase: Requires labeled data

Expectation: If SFT phase is beneficial due to mitigating vanishing gradients for RFT

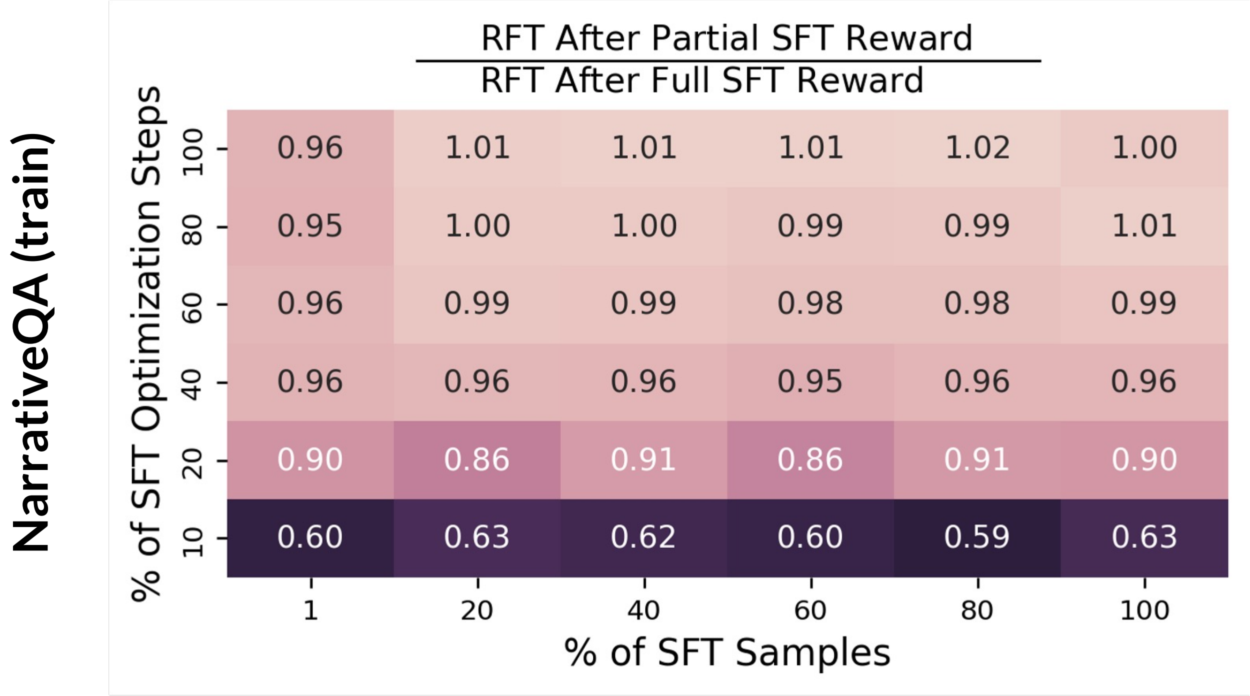
→ A few steps of SFT on small # of labeled samples should suffice

A Few SFT Steps on a Small Number of Samples Suffice

Limitation of Initial SFT Phase: Requires labeled data

Expectation: If SFT phase is beneficial due to mitigating vanishing gradients for RFT

→ A few steps of SFT on small # of labeled samples should suffice

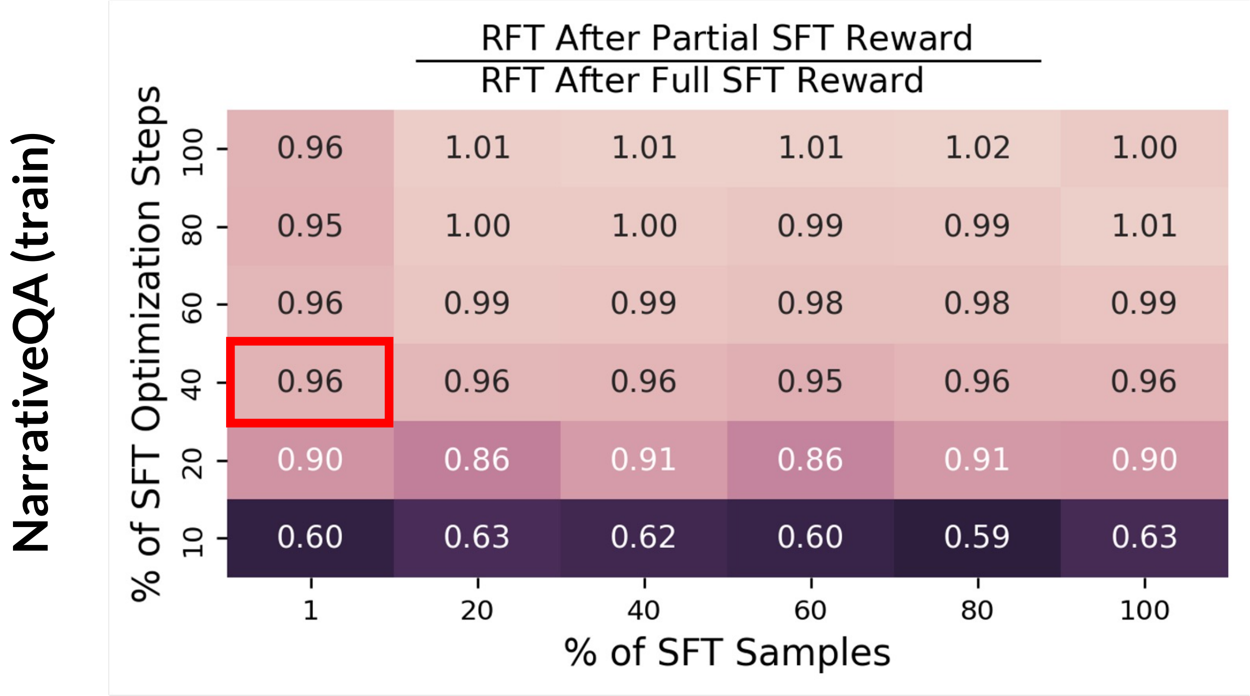


A Few SFT Steps on a Small Number of Samples Suffice

Limitation of Initial SFT Phase: Requires labeled data

Expectation: If SFT phase is beneficial due to mitigating vanishing gradients for RFT

→ A few steps of SFT on small # of labeled samples should suffice

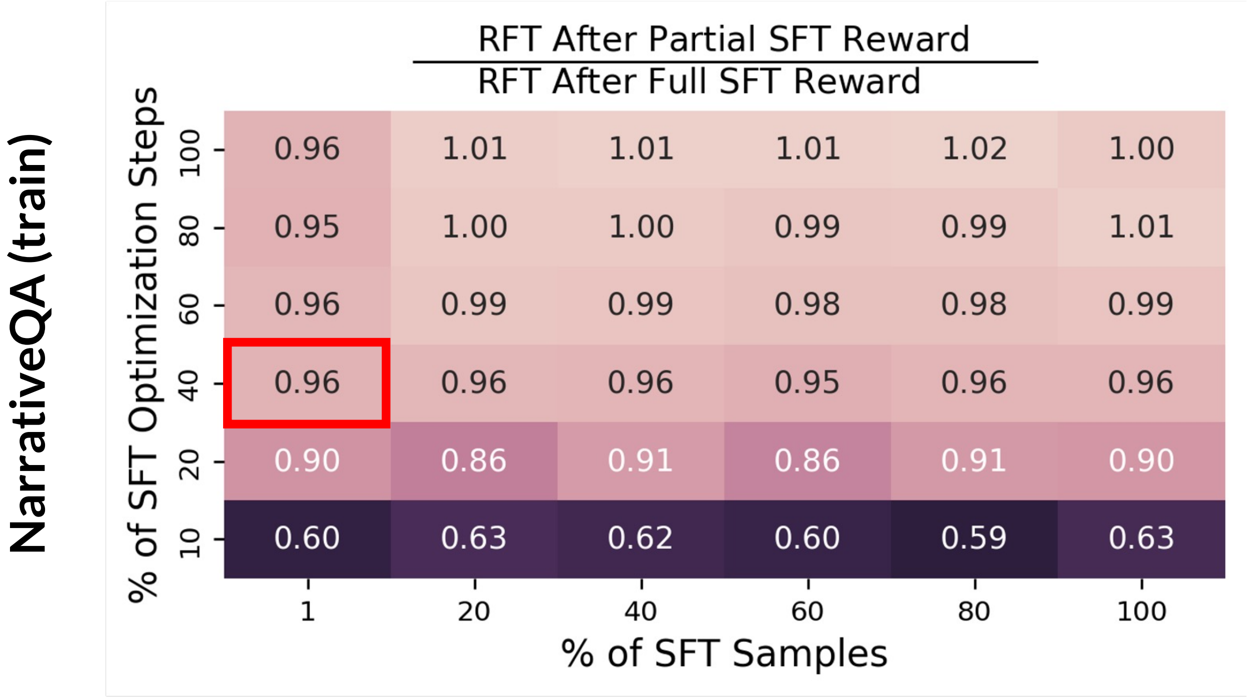


A Few SFT Steps on a Small Number of Samples Suffice

Limitation of Initial SFT Phase: Requires labeled data

Expectation: If SFT phase is beneficial due to mitigating vanishing gradients for RFT

→ A few steps of SFT on small # of labeled samples should suffice



The initial SFT phase does not need to be expensive!

Conclusion: Vanishing Gradients in RFT

Conclusion: Vanishing Gradients in RFT

$\nabla_\theta \mathbf{V}_\theta(\mathbf{x}) \approx \mathbf{0}$

Expected gradient for an input vanishes in RFT
if the input's reward std is small

Conclusion: Vanishing Gradients in RFT

$\nabla_\theta \mathbf{V}_\theta(\mathbf{x}) \approx \mathbf{0}$ **Expected gradient for an input vanishes in RFT** if the input's reward std is small

Experiments + theory: **vanishing gradients in RFT are prevalent and detrimental** to maximizing reward

Conclusion: Vanishing Gradients in RFT

$\nabla_\theta \mathbf{V}_\theta(\mathbf{x}) \approx \mathbf{0}$ **Expected gradient for an input vanishes in RFT** if the input's reward std is small

Experiments + theory: **vanishing gradients in RFT are prevalent and detrimental** to maximizing reward

Initial SFT phase allows overcoming vanishing gradients in RFT, and **does not need to be expensive**

Conclusion: Vanishing Gradients in RFT

$\nabla_\theta \mathbf{V}_\theta(\mathbf{x}) \approx \mathbf{0}$

Expected gradient for an input vanishes in RFT
if the input's reward std is small

Experiments + theory: **vanishing gradients in RFT are prevalent and detrimental** to maximizing reward

Initial SFT phase allows overcoming vanishing gradients in RFT, and **does not need to be expensive**

↓

Reward std is a key quantity to track for successful RFT

Thank You!

Work supported by:
Apple scholars in AI/ML PhD fellowship, Google Research Scholar Award, Google Research Gift, the Yandex Initiative in Machine Learning, the Israel Science Foundation (grant 1780/21), Len Blavatnik and the Blavatnik Family Foundation, Tel Aviv University Center for AI and Data Science, and Amnon and Anat Shashua