Conclusion Form Example

Paper	Date	I Area	II Focus	III Sub-topic	Paper Type	2 Problem/Target	2.1 Background	2.2 What causes the P	2.3 Importance of P	3 Solution	3.1 Method They Used	3.2 Method Types	4 Old Method Compared	4.1 Advantages	4.1+ Adv	4.2 my questions	4.2+ my questions	5 Support for result	6 Best Method I think	7 Possible Improvement	7.1 Improve-Source
@Featgraph: A flexible and efficient backend for graph neural network systems	@2020/09/29	ML	GNN	System	New Model	existing graph processing systems fail to exploit locality and parallelism	1. Sparse operation matters in GNN workloads. 2. Existing systems **does not support feature dimension**, and **rarely exploit parallelism.**	under explored		FeatGraph: **decomposing** a kernel specification into sparse templates and UDFs; **co-optimizing** graph traversal and feature dimension computation	**decompose**: SpMM & SDDMM templates + UDFs; **optimization**: graph partitioning + user specified optimization for UDFs e.g. feature dimension schedule (FDS)	CS optimization Math Theory	traditional graph processing systems: Ligra on CPU and Gunrock on GPU; vendor-provided sparse libraries: MKL on CPU and cuSPARSE on GPU	Efficiency Flexibility				Experiment	Featgraph	Not familiar with this topic, but will it cost more memory?	
@From Local to Global: Spectral-Inspired Graph Neural Networks	@2022/11/04	ML	GNN	message-passing algorithms (MPNNs)	New Model	Existed MPNNs suffer from miss long-range signals, perform poorly on some heterophilous graphs, over-smoothing, over-squashing; Existed optimization methods are not well-understood theoretically or increase the overall computational complexity	Popular GNNs are message-passing algorithms (MPNNs) that aggregate and combine signals in a local graph neighborhood.	*design deficiency?*		**PowerEmbed** — a simple layer-wise normalization technique to boost MPNNs	**augmenting** the message-passing layer with **a simple normalization step** **couple PowerEmbed with an inception network**	STAT Theory	spectral methods unsupervised methods: SGC & SIGN semi-supervised methods: GCN, GAT, Geom-GCN, GCNII, GPR-GNN, JK-Concat	Performance	*higher accuracy, simple structure, rich experiments, rich comparison methods	unstable performance	No solve all problems: complexity; Not a single setting of the model outperforms all others; Not achieve best performance on homo graphs	Experiment	Power series (on hetero) Semi-supervised (on homo)	stabilize its performance so it can only use one-setting to outperform other models; extend its ability so it could perform the best on homophilous graphs; figure out why normalization technique causes poor performance on homophilous graphs	
@❤️ A LOCAL GRAPH LIMITS PERSPECTIVE ON SAMPLING-BASED GNNS	@2023/10/17	ML	GNN	large graph	Optimizations	Large-graph sampling suffers from **training inefficiencies and large memory requirements**, while the existing methods **lack theory support**.		under explored		propose a **theoretical framework** to sample the graph, and offers **a simplified training procedure** to learn parameters of the whole through small subgraphs, and **prove**	**node sampling** **computational graph sampling** **Graph limit theory:** Benjamini-Schramm convergence	CS optimization STAT Theory	GCN, GraphSAGE original	Efficiency Theory	Provide theory support; Invent new efficient techs; **Extending GNN convergence results** to many well-known architectures	unstable performance	Performance unstable: sometimes perform bad than the full graph	Experiment Proof	This one	stabilize its performance extend it to broader aspect of GNN	
@Topological and Temporal Data Augmentation for Temporal Graph Networks	@2023/01/01	ML	GNN	temporal	Optimizations	**existing data augmentation strategies for temporal graphs, especially continuous-time dynamic graphs (CTDGs), are largely heuristic and hand-crafted**, which may **alter the inherent semantics of temporal graphs**, thereby degrading the performance of downstream tasks.		**alter the inherent semantics of temporal graphs** data redundancy [20, 21] along the temporal axis ⇒ overfit		**topological and temporal data augmentation (TTDA) techniques**, on the representation space of messages functions instead of altering original temporal graph structure or timestamps.	**topological strategy: update memory module by messages from previous batch** **maximize the mutual information** between z˜i(t) and **original node embedding** zi(t). **temporal strategy:** **impose a smoothness constraint over time** **maximize the mutual information between temporal augmented projection** p˜i(t) **and the original projection** pi(t)	Data augmentation	JODIE [11], TGAT [22], DyRep [18], TGN [16], MeTA [19]	Performance	*higher accuracy	lack of proof time memory cost unstable performance	linear loss function time memory cost lack of proof	Experiment	TTDA while sometimes MeTA?	better loss function considering correlations between topology and temporary Why MI is the best way to enhance performance?	