FastGCN (Iclr 2018)

Main contributions
- They interpret graph convolutions as integral transforms of embedding functions under probability measures.
- Then they utilize Monte Carlo approaches to consistently estimate the integrals.

Advantages
- Sampling helps to utilize important nodes/information sharing, not all of those. It will accelerate the convergence.
- Scale: 3000-20,000 nodes, 5000-1160,0000 edges. Advantages is enlarged with scaling increased.
- We can also generalize this method to many graph models whose formulations are based on first-order neighborhoods.

Drawbacks
- Essence: Importance sampling, where variance reduction must be considered.

Inductive Representation Learning on Graphs (nips 2018)

Motivation
- Considering unseen nodes.
- Utilizing underlying information sharing better.

Main contributions
- Learn a function that generates embeddings by sampling and aggregating features from a node's local neighborhood, instead of training individual embeddings for each node.
- Learning + sampling, are replaced the attention structure, to avoid vast amounts of parameter learning.

Other notes
- It helps prediction problems.
- Speed improvements, in large-scale problem.
- Drawbacks: generalization and transfer abilities need to be checked.

Learning Permutations with Sinkhorn Policy Gradient (SPG)

Main Contributions: End to end work for demonstration learning.
- Learn representations from high-structured action space, with a Sinkhorn-layer
- Motivated by Sinkhorn Iteration/decomposition for optimal transport: a best way to measure two probabilities. Connection to Wasserstain distance.
- Actor-critic architecture can be trained end-to-end.

Other Notes
- Revisit demonstration from new measurements.
- Sinkhorn layer makes the discrete space to be approximated by a continuous one.

Boosting Combinatorial Problem Modeling with Machine Learning (ijcai 2018)

Motivation
- Learning for modeling. (A good formulation of MIP maybe LP.)
  - Create expert models by given methods.
  - Refine according to expert.

Methods
- Learning via a Native Constraint Language
  - From iterative interactions between a domain and an optimization expert
  - Picking constraints from a set of potential (instantiated) candidates (called a bias) and adding them to a target constraint network
  - Training set is online update.
- Incorporate a fully-fledged ML model within a combinatorial optimization model
  - Constraints are replaced by a ML model
  - Parts of solvers are encoded into a ML model
    - BB and tree
    - Cutting planes
- ML Models as Problem Models
  - An end-to-end way
  - ML surrogates for original problem

Our Focus
- where is replaced by ML ?
- discrete -> continuous?
- Sampling helps to simplify the problem?
- Online update needs sampling?