Diversity Based Edge Pruning of Neural Networks Using Determinantal Point Processes

Rupam Acharyya[1], Boyu Zhang*[2], Ankani Chattoraj*[3], Shouman Das[4], Daniel Stefankovic[2]
Mathematics Department, University at Buffalo[1],
Department of Computer Science, University of Rochester[2],
Department of Brain & Cognitive Science, University of Rochester[3],
Department of Mathematics, University of Rochester[4]

Introduction

- **Neural Network Pruning:** Reduce the size of the network without degrading its performance much.
- **Motivation:** Pruning can help reducing the time complexity (of fine tuning) and space complexity with pre-trained networks containing billions of parameters (e.g. BERT).
- **Limitation:** Lots of pruning methods available, but why do they work?
- **This Work:** Takes a step towards explaining pruning performance.

Preliminaries

Determinantal Point Process (DPP)

DPP is a probability distribution defined on the set of subsets of a ground set $(\mathscr{U})$. For $Y \subset \mathscr{U}$,

$$\mathbb{P}[\mathbf{Y} = Y] = \frac{\det(L_Y)}{\det(L+I)}$$

where L is the *kernel matrix* and L_Y is the submatrix defined by the rows and columns indexed by Y.

$$\mathcal{G} = \{\underbrace{(2,4)}_{y_1}, \underbrace{(1,0)}_{y_2}, \underbrace{(1,2)}_{y_3}\}$$

$$\underbrace{\begin{bmatrix} 2 & 4 \\ 1 & 0 \\ 1 & 2 \end{bmatrix}}_{U} \cdot \underbrace{\begin{bmatrix} 2 & 1 & 1 \\ 4 & 0 & 2 \end{bmatrix}}_{U^T} = \underbrace{\begin{bmatrix} 20 & 2 & 10 \\ 2 & 1 & 1 \\ 10 & 1 & 5 \end{bmatrix}}_{L}$$

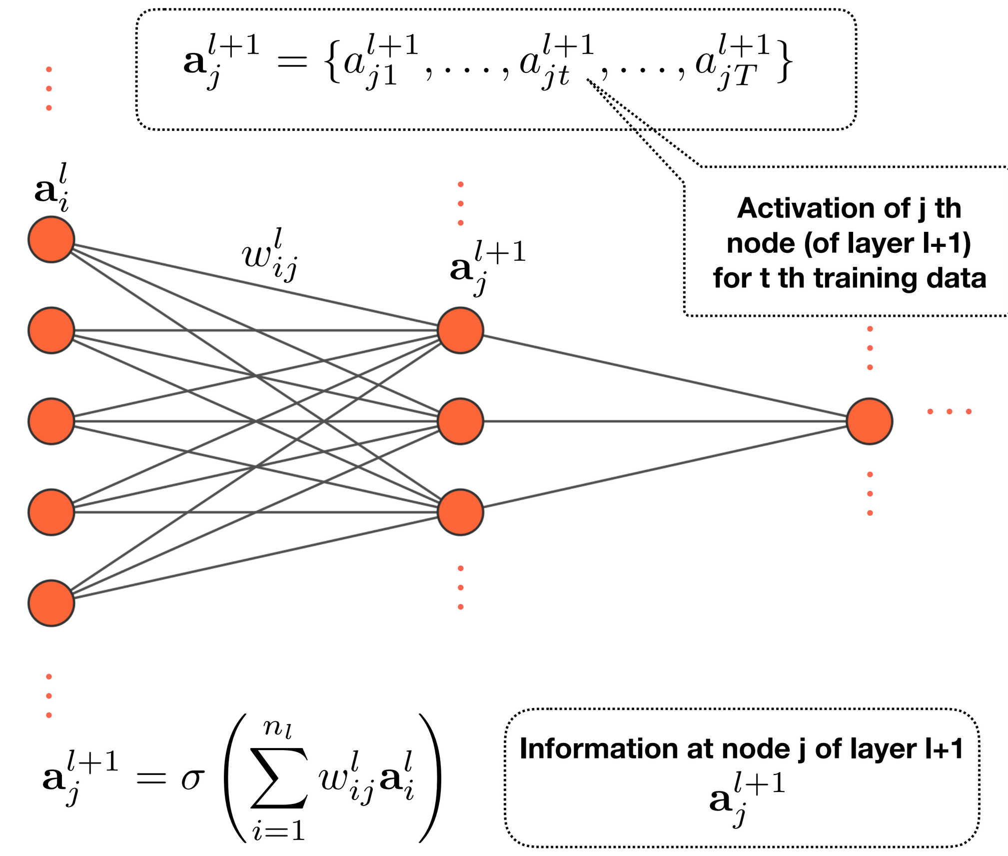
$$\mathbb{P}[Y = \{y_1, y_3\}] \propto \det \begin{bmatrix} 20 & 2 & 10 \\ 2 & 1 & 1 \\ 10 & 1 & 5 \end{bmatrix} = \mathbf{0}$$

$$\mathbb{P}[Y = \{y_1, y_2\}] \propto \det \begin{bmatrix} 20 & 2 & 10 \\ 2 & 1 & 1 \\ 10 & 1 & 5 \end{bmatrix} = \mathbf{16}$$

Previous Work

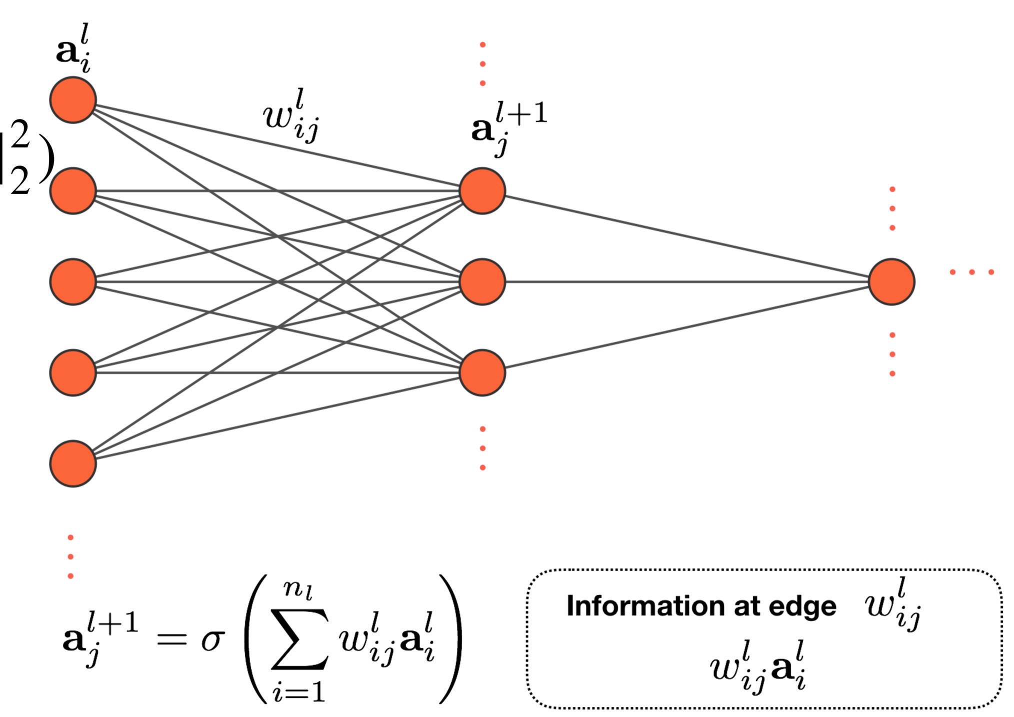
Node Pruning using DPP-DIVNET (Mariet et al.)

- Idea:
  - Sample a subset of nodes for each layer using the DPP defined by the kernel matrix defined as above.
  - Later some *re-weighting* of the edges is needed to compensate for the lost nodes (can be done efficiently).
- Need:
  - Some representation of node: $\mathbf{a}_j^{l+1} = \{a_{j1}^{l+1}, \ldots, a_{jt}^{l+1}, \ldots, a_{jT}^{l+1}\}$
  - A kernel matrix $L_{jk} = \exp(-\beta\,||\mathbf{a}_j^l - \mathbf{a}_k^l||_2^2)$



Diversity based Edge Pruning of Neural Network

- **Kernel for DPP:**
  - The kernel matrix is defined as: $L_{jk}^i = \exp(-\beta\,||w_{ij}^l \mathbf{a}_i^l - w_{ik}^l \mathbf{a}_k^l||_2^2)$
  - For layer l there will be n_l kernels $L^1, \ldots, L^{n_l}$
- **Method:**
  - Keep a subset of edges for each layer using the DPP defined by the above kernel matrix.
  - Re-weighting of the edges is needed to compensate for the lost edges

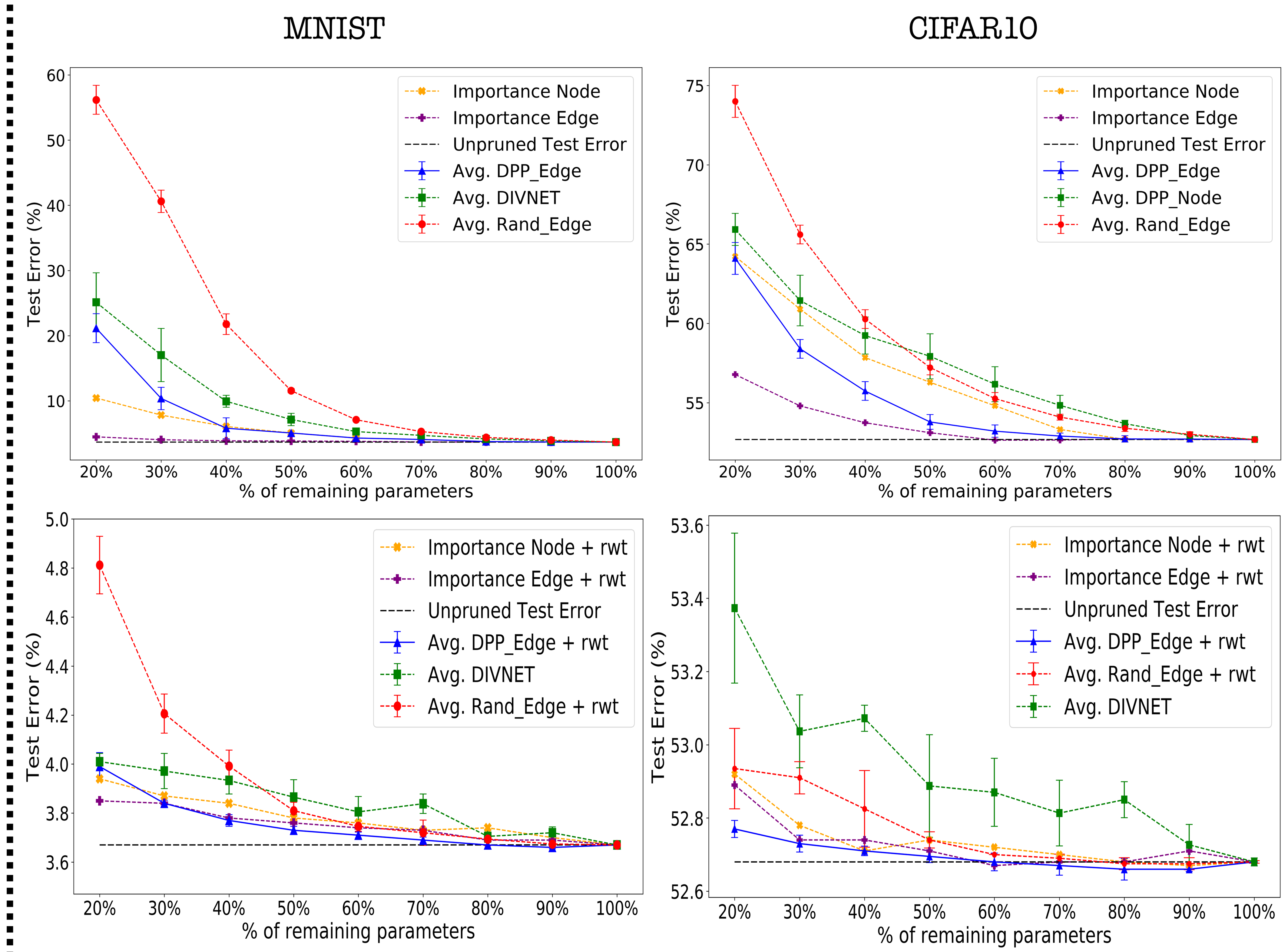


- **Retrieving Pruned Information by Reweighting**
  - $S_j = \{i_1, \ldots, i_{k_e}\}$:= the set of incoming edges chosen for the node v_j^l.
  - $\hat{w}_{ij}^{l-1} = w_{ij}^{l-1} + \delta_{ij}^{l-1}$:= the new weight of the edges after reweighting.
  - To minimize the lost information after pruning, we minimize:

$$\left|\left| \sum_{i=1}^{n_{l-1}} w_{ij}^{l-1}\mathbf{a}_i^{l-1} - \sum_{i \in S_j} \hat{w}_{ij}^{l-1}\mathbf{a}_i^{l-1} \right|\right|_2 = \left|\left| \sum_{i \in \bar{S}_j} w_{ij}^{l-1}\mathbf{a}_i^{l-1} - \sum_{i \in S_j} \delta_{ij}^{l-1}\mathbf{a}_i^{l-1} \right|\right|_2$$

Result

- When reweighting is not applied, importance edge pruning performs the best in both the datasets.
- DPP edge pruned network significantly outperforms all other pruning methods when reweighting is added.
- Importance node pruning with reweighting performs better than DIVNET, which was not explored in Mariet et al.
- DPP edge pruned network generalizes better than the unpruned dense network for both the datasets (see at 90\% for `MNIST` and 70-90\% for `CIFAR10`).



Conclusion & Future Work

- Extend for feed-forward networks with more than two layers.
- Explore in other neural network architectures.
- Can DPP-Edge find winning tickets of Lottery Ticket Hypothesis?

Reference:

1) Zelda Mariet and Suvrit Sra. Diversity networks: Neural network compression using determinantal point process.