Towards Device Efficient Conditional Image Generation
Nisarg A. Shah and Gaurav Bharaj
AI Foundation, California, USA

**Motivation**
- We propose a two-stage novel strategy where, first, we condense the channel weights, such that, as few channels are used.
- Later we prune, nearly zeroed out weight activations, and fine-tune the autoencoder.
- To maintain image quality, fine-tuning is done via student-teacher training, condensed model - (Teacher)

**Goal**: Efficient and accurate representation of model weights, for superior channel pruning

**Our Approach**
- We propose channel weight and layer device performance device regularisation, both operating at intra and inter-layer level
- Channel importance factor $\gamma_i$ is equivalent to magnitude of the weights of the corresponding channels.
- We calculate the run-time for each layer across a particular device, and use it as a multiplicative factor $l(i)$ for that layer. (device-dependent)

**Method Overview**
- We propose, channel weight and layer device performance device regularisation, both operating at intra and inter-layer level
- Channel importance factor $\gamma_i$ is equivalent to magnitude of the weights of the corresponding channels.
- We calculate the run-time for each layer across a particular device, and use it as a multiplicative factor $l(i)$ for that layer. (device-dependent)

**Network Architecture**
- **Stage I**: DEIG U-net
- **Stage II**: DEIG U-net
- **Stage III**: DEIG U-net

**‘Hinge’ based pruning**
- On Stage I training, a model with a considerable amount of near zero weight channels are obtained with considerable distinction.
- The inclination point that shows the threshold between these two types of channels is identified as the "hinge".
- In turn, not requiring to take an arbitrary guess or a global threshold on the number of channels to be pruned.

**Channel level**:
$$L_i = \sum_{j=1}^{N} f(j) \cdot |W_{i,j}|$$
- $W_{i,j}$ = Filter weight of $i^{th}$ layer and $j^{th}$ sorted channel
- $f(j)$ = channel regularisation function; Linear, Uniform, ...

**Layer level**:
$$L_{\text{PENAL}} = \sum_{i=0}^{N} l(i) \cdot L_i$$
- $l(i)$ = Runtime for layer on particular device

**Experiments and Results**
- Generator models such that UNet and ResUNet, and there corresponding DEIG versions were used
- Inference times are calculated for both CPU and GPU
- Conditional image generation tasks - Segmentation mask to images, images to cartoonization and CycleGAN