DECODING EFFICIENT DEEP LEARNING:
PATH TO SMALLER, FASTER, AND BETTER MODELS

RESEARCH PAPER: https://arxiv.org/pdf/2106.08962.pdf, Google Research, US

PRESENTED BY: Somya Mishra, San Jose State University

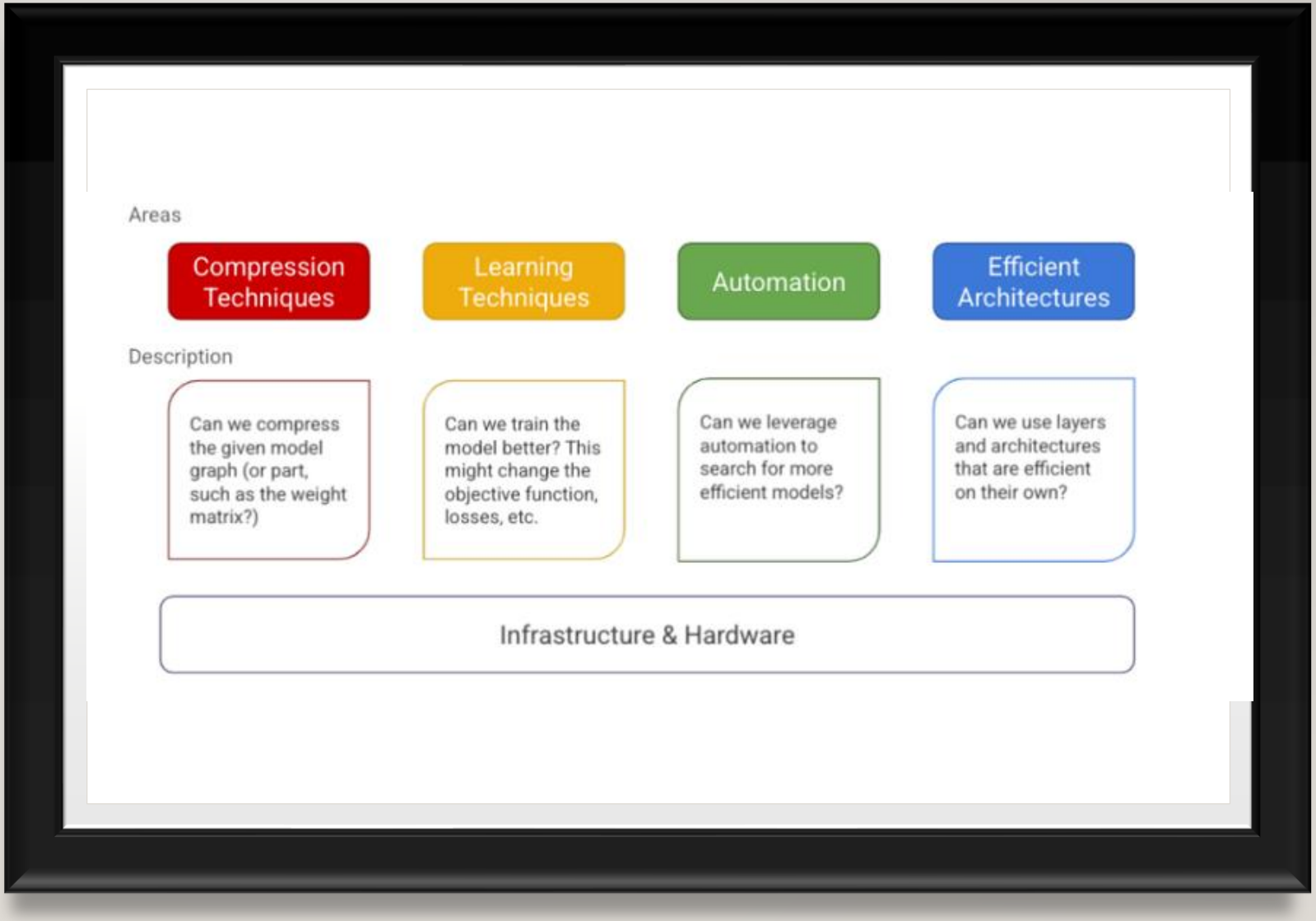
WHY DO WE NEED EFFICIENT DEEP LEARNING?

- Machine Learning and Artificial Intelligence at large is evolving at a tremendous pace!
- Every researcher is trying to achieve the best models and beat benchmarks continuously.
- When a model is deployed, a lot of focus must be spent on analyzing whether deep learning models can be efficiently scaled for people who might not have millions of dollars to train the models and gigantic machines to deploy their models.
- How does one make efficiency related decisions?
- Quality vs Footprint tradeoff

WHAT ARE THE CHALLENGES DURING MODEL TRAINING AND DEPLOYING?

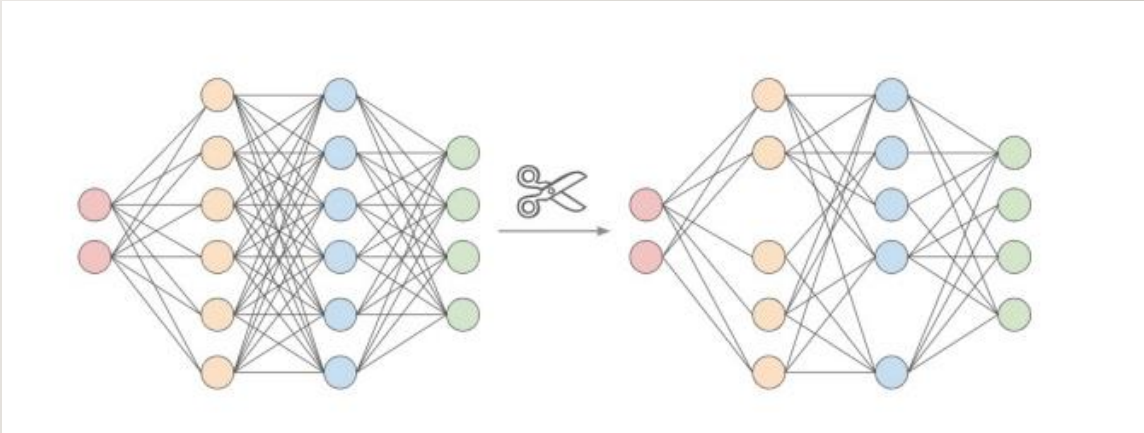
- Sustainable Server-Side Scaling
- Enabling On-Device Deployment
- Privacy & Data Sensitivity
- New Applications
- Explosion of Models

A MENTAL MODEL



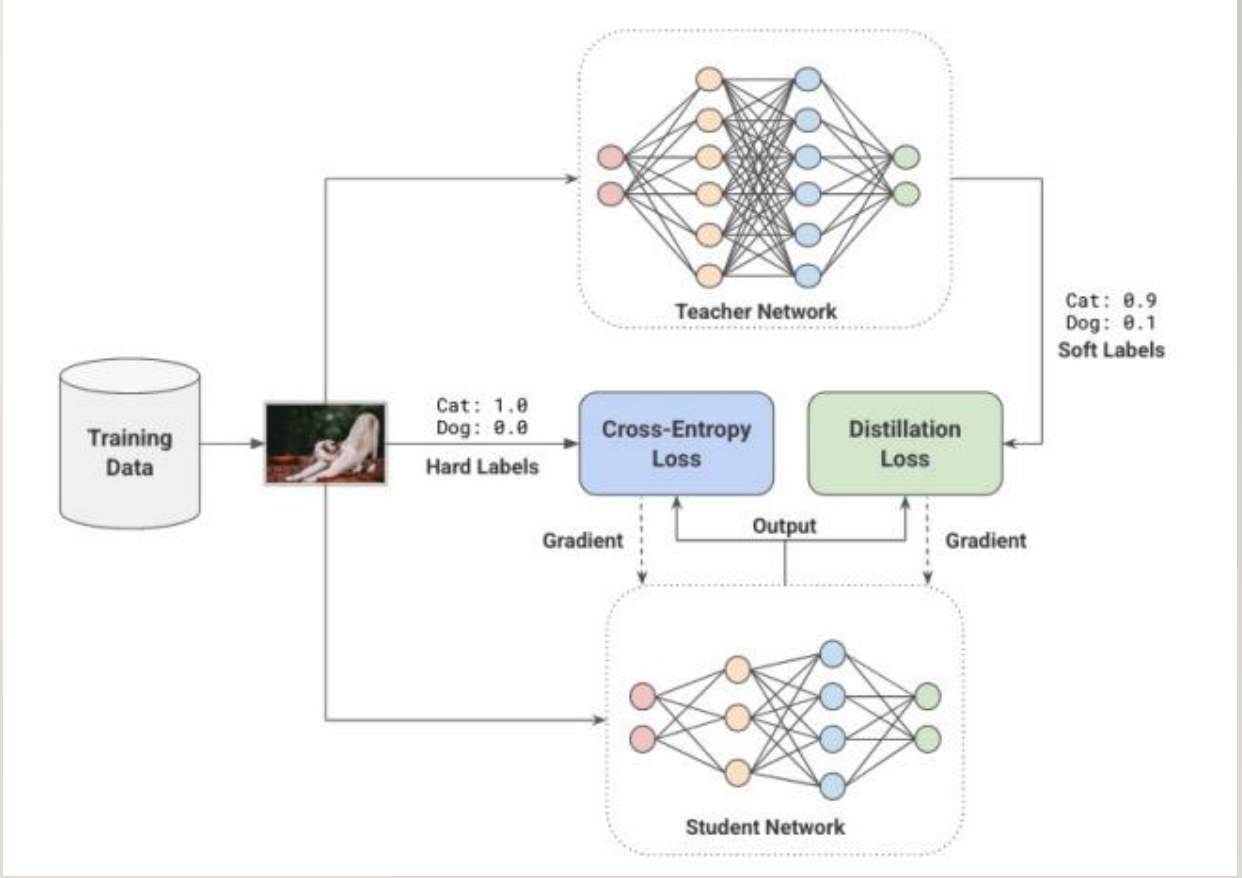
COMPRESSION TECHNIQUES

- The goal is to see whether a big model can be converted to a small model , with the end goal being to be deployed on an edge device.
- Quantization is a compression technique: reducing the weight metric of a layer by reducing its precision.



LEARNING TECHNIQUES

- Can be considered as a replacement to traditional supervised learning algorithms.
- The goal is to train a model differently to achieve better quality metric accuracy, F1 score, precision, recall, etc.
- Achieve same baseline quality with a smaller model, with a trade off between quality and number of parameters / layers.



AUTOMATION

- Tuning hyper parameters to improve accuracy, which could then be exchanged for a model with lesser parameters.
- Train-test-split method can be used to initialize the parameters and observe the score/accuracy for each parameter.

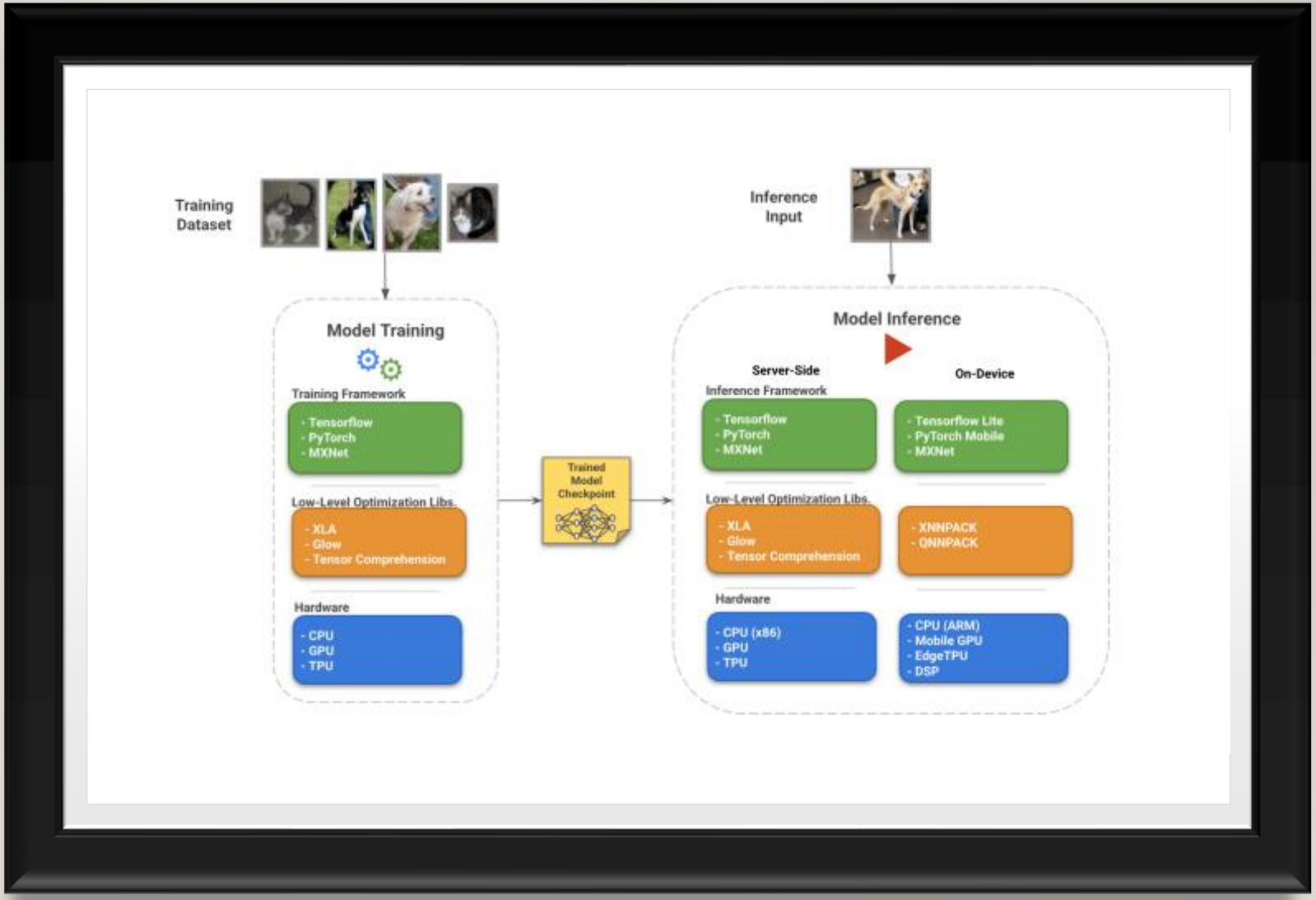
EFFICIENT ARCHITECTURES

- Basic building blocks that are designed from scratch such as convolutional layers and attention layers, that are a significant leap over the baseline methods used before.
- Directly improve the efficiency gains.

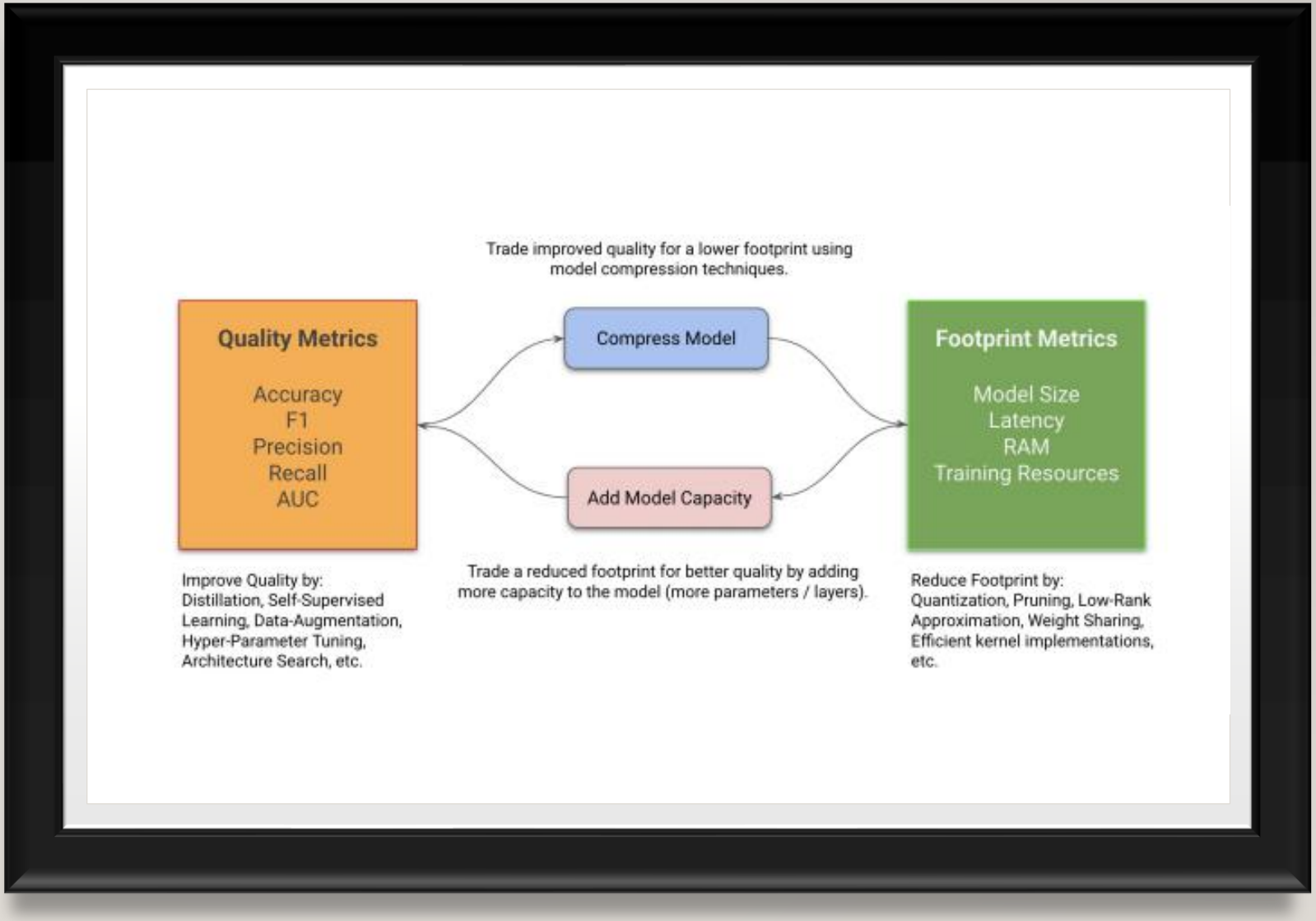
INFRASTRUCTURE

- Robust combination of good software and hardware
- First step: Model training and second step: Model Inference
- Hardware such as GPU and TPU can be used for speeding up linear algebra operations!

VISUALIZATION OF HARDWARE AND SOFTWARE INFRASTRUCTURE WITH EMPHASIS ON EFFICIENCY



PRACTITIONER'S GUIDE TO EFFICIENCY:
TRADE OFF BETWEEN MODEL QUALITY AND FOOTPRINT



SHRINK-AND-IMPROVE FOR FOOTPRINT-SENSITIVE MODELS

- Followed in cases where one wants to reduce the footprint and keep the quality the same for on-device deployments and server-side model optimization

GROW-IMPROVE-AND-SHRINK FOR QUALITY-SENSITIVE MODELS

- Followed in cases where one wants to deploy models that have better quality while keeping the same footprint.

EXAMPLES OF TECHNIQUES TO USE IN GROW, SHRINK, AND IMPROVE PHASES:

Grow (Model Capacity)	**Shrink** (Footprint)	**Improve** (Quality)
Add layers, width, etc. either manually or using width / depth / compound scaling multipliers	Reduce layers, width, etc. either manually or using width / depth / compound scaling multipliers	Manual Tuning (Architecture / Hyper-Parameters / Features, etc.)
	Compression Techniques: Quantization, Pruning, Low-Rank Factorization, etc.	**Learning Techniques**: Data-Augmentation, Distillation, Unsupervised Learning, etc.
	Automation: Hyper-Param Optimization, Architecture Search, etc.	**Automation**: Hyper-Param Optimization, Architecture Search, etc.
	Efficient Layers & Models: Projection, PQRNN, (NLP), Separable Convolution (Vision), etc.	**Efficient Layers & Models**: Transformers (NLP), Vi-T (Vision), etc.

CONCLUSION

- First, we need to achieve a new pareto-frontier using the efficiency techniques.
- Second, we demonstrate tradeoffs for both 'Shrink-and-Improve', and 'Grow-Improve-and-Shrink' strategies.
- Last, we provide empirical evidence that it is possible to either reduce model capacity to bring down the footprint (**shrink**) and then recover the model quality that they traded off (improve) or increase the model capacity to improve quality (**growing**) followed by model compression (shrinking) to improve model footprint!

REFERENCE

- Menghani, G. (2021). Efficient Deep Learning: A Survey on Making Deep Learning Models Smaller, Faster, and Better. *arXiv preprint arXiv:2106.08962*.

THANK YOU!