A Survey of Model Compression Methods

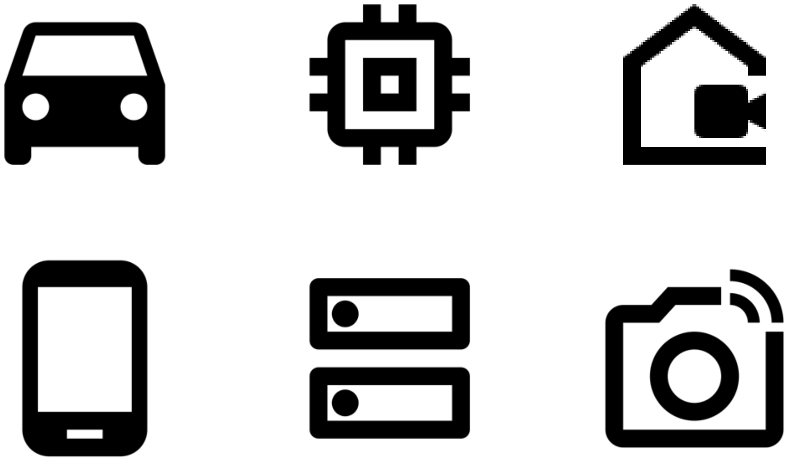
Rustem Feyzkhanov
Staff Machine Learning Engineer
Instrumental

Agenda

- Cloud-to-edge ML pipeline and optimizations
- Model serialization
- Pruning
- Quantization
- Weight clustering
- Knowledge distillation
- Architecture search
- Summary

Challenges with ML on Edge

- Latency
- Memory usage
- Energy usage
- Disk usage



Cloud-to-edge ML pipeline

Data gathering → Architecture search → Model training → Model compilation → Deployment and model inference

Data gathering
- Data acquisition
- Data labeling

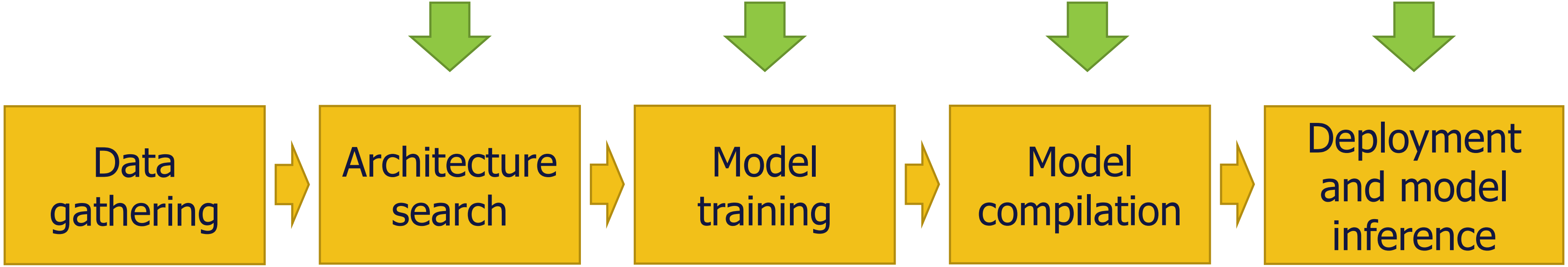
Architecture search
- Feature selection
- Model architecture

Model training
- Cloud training pipeline
- Parameter optimizations

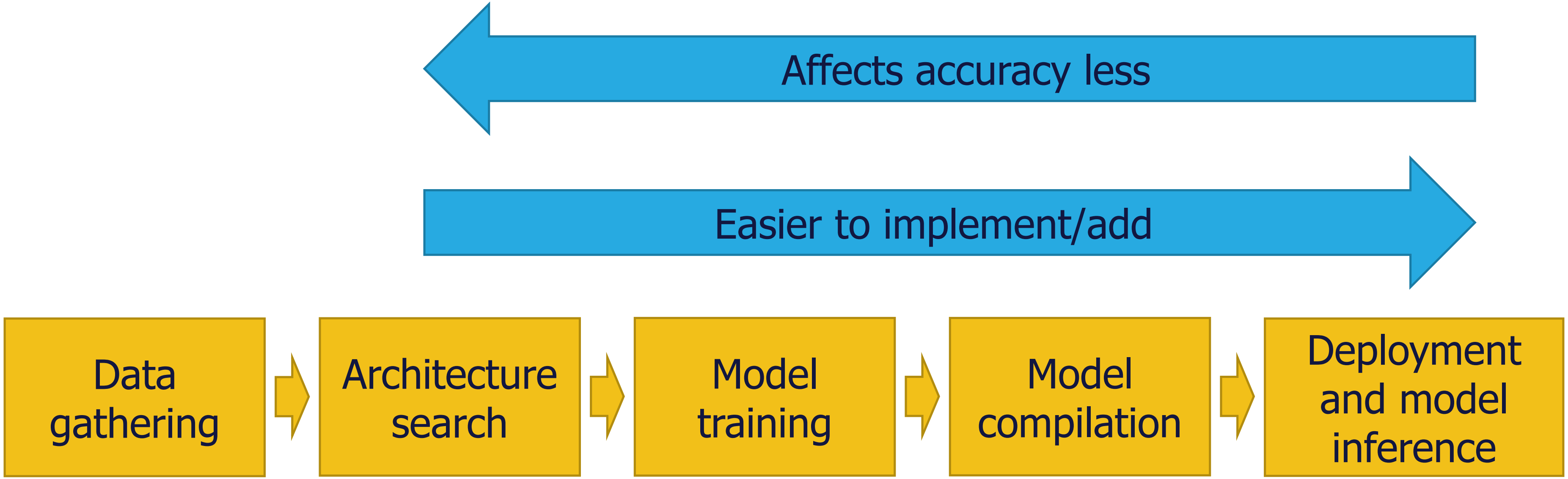
Model compilation
- Serialization
- Post training optimizations
- Compiling model in cloud for hardware

Deployment and model inference
- Serving model to the edge
- Initializing model
- Making predictions

Optimizations in ML pipeline



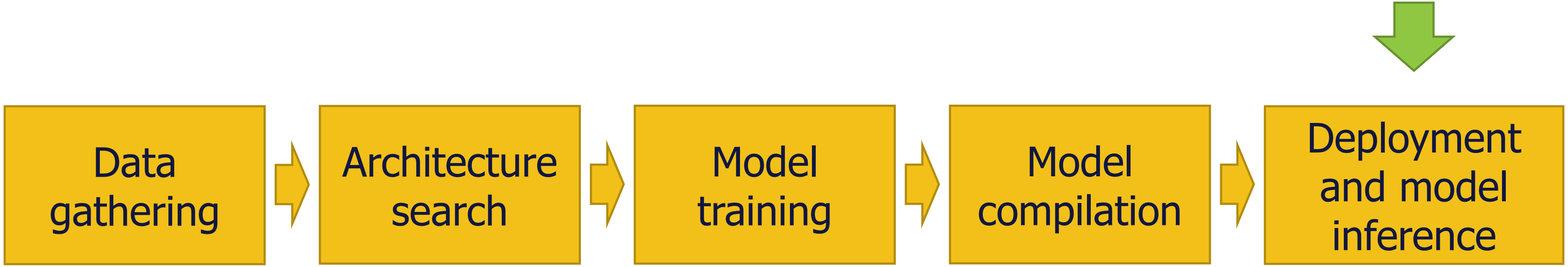
Optimizations in ML pipeline



Model serialization

Inference frameworks and model serialization

- Hardware specific optimizations
- Framework built-in native model compression methods
- Significantly affects speed



Inference frameworks

Frameworks focused on running inference. Lightweight, focused on specific hardware, require separate serialization.

- Hardware agnostic: e.g., TorchScript, ONNX, TFLite*
- Hardware specific:
  - CPU: e.g., OpenVINO (Intel)
  - GPU: e.g., TensorRT
  - Mobile: e.g., CoreML
  - NPU: Check out Embedded Vision Alliance members :-)

Inference frameworks - example

- Object detection - YOLOv5s

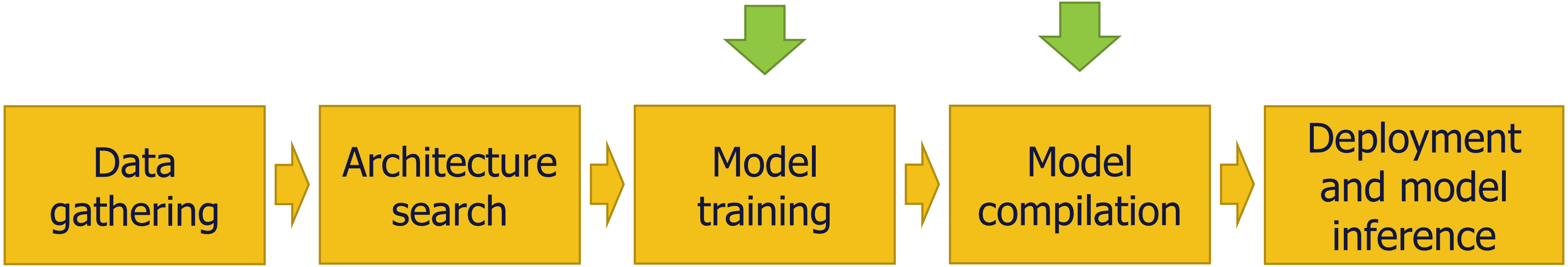
Framework	Size, MB	CPU inference, ms	GPU (V100) inf, ms
PyTorch	29.5	127.61	10.19
TorchScript	29.4	131.23	6.85
TensorRT	33.3	N/A	1.89
ONNX	29.3	69.34	14.63
OpenVINO	29.3	66.52	N/A
TFLite	29.0	316.61	N/A

From https://docs.ultralytics.com/yolov5/tutorials/model_export/

Pruning

Pruning

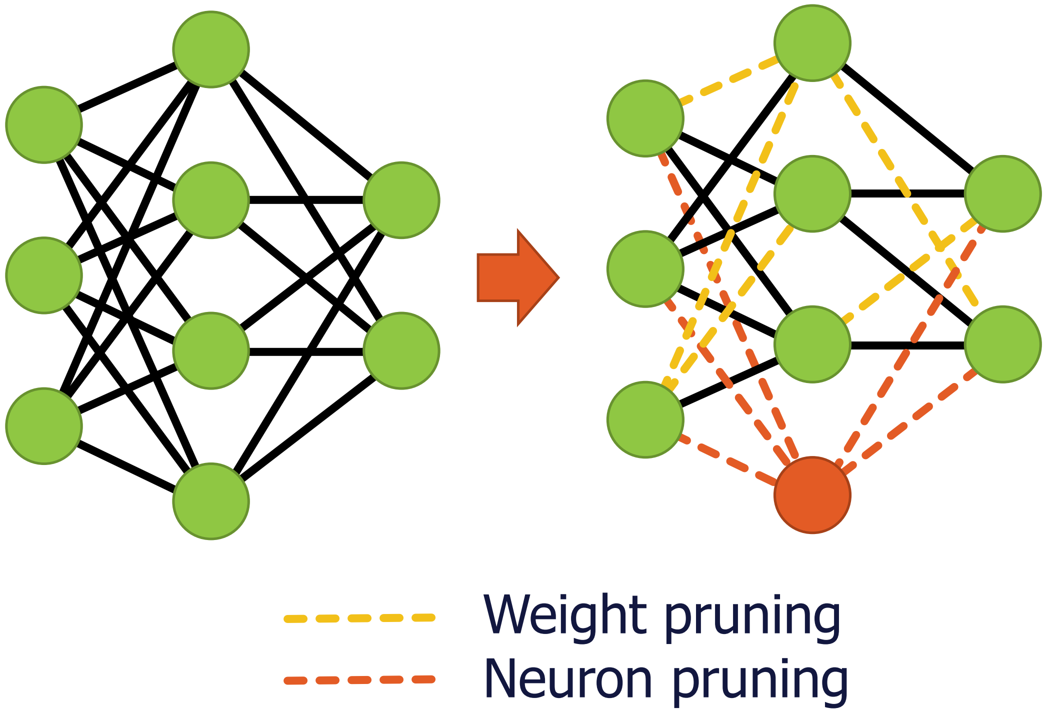
- Eliminating redundant or unimportant parameters
- Affects accuracy, but can be addressed by finetuning
- Affects size more than speed



INSTRUMENTAL

Types of pruning

- Weight pruning
- Structural pruning
  - Neuron
  - Layer
  - Filter
  - Channel



Weight pruning example

- Image classification example – weight pruning

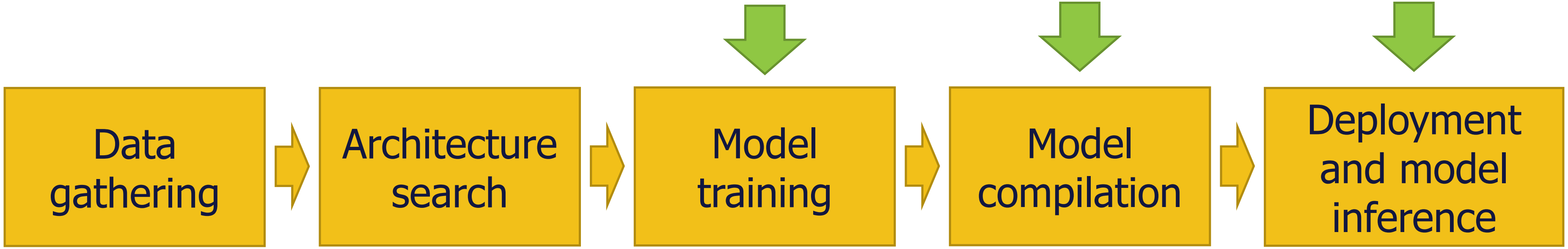
	Configuration	Number of parameters	Top-1 accuracy, ImageNet
InceptionV3	Original	27.1M	78.1%
	50% sparsity	13.6M	78.0%
	75% sparsity	6.8M	76.1%
	87.5% sparsity	3.3M	74.6%

From https://www.tensorflow.org/model_optimization/guide/pruning

Quantization

Quantization

- Reducing numerical precision of weights and activations
- Affects accuracy, but can be addressed during training
- Checkout the talk "**Practical Approaches to DNN Quantization**" later today



Quantization example

- Image classification example – quantization

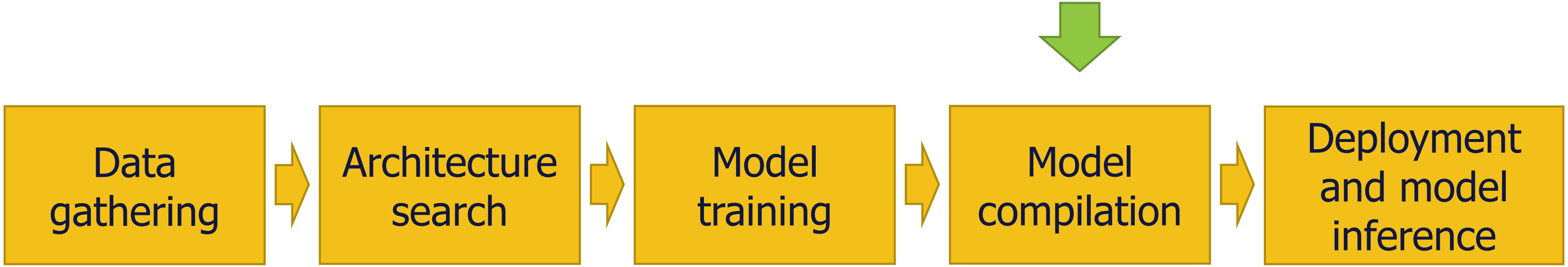
	Configuration	Size, MB	Inference (CPU), ms	Top-1 accuracy, ImageNet
InceptionV3	Original	95.7	1130	78.1%
	Post training quantization	23.9	845	77.2%
	Quantization aware training	23.9	543	77.5%

From https://www.tensorflow.org/lite/performance/model_optimization

Weight clustering

Weight clustering

- Cluster model weights and use indices
- Only optimizes model size
- Similar to quantization, but doesn't change computation complexity



Weight clustering example

- Replaces weights with reference to the closest centroids
- Centroids are usually rounded, but not quantized so main gain is model size

Weight matrix

0.86	-1.88	-1.44
-1.7	1.58	0.12
1.9	0.46	1.37

1	2	2
2	3	1
3	1	3

Centroid	Index
0.33	1
-1.26	2
1.92	3

Weight clustering example

- Image classification example

	Configuration	Size, MB	Top-1 accuracy, ImageNet
MobileNetV2	Original	12.38	71.7%
	Last 3 layers, 32 clusters	7.03	70.9%
	Last 3 layers, 16 clusters	6.68	70.7%
	All layers, 32 clsuters	4.05	69.7%

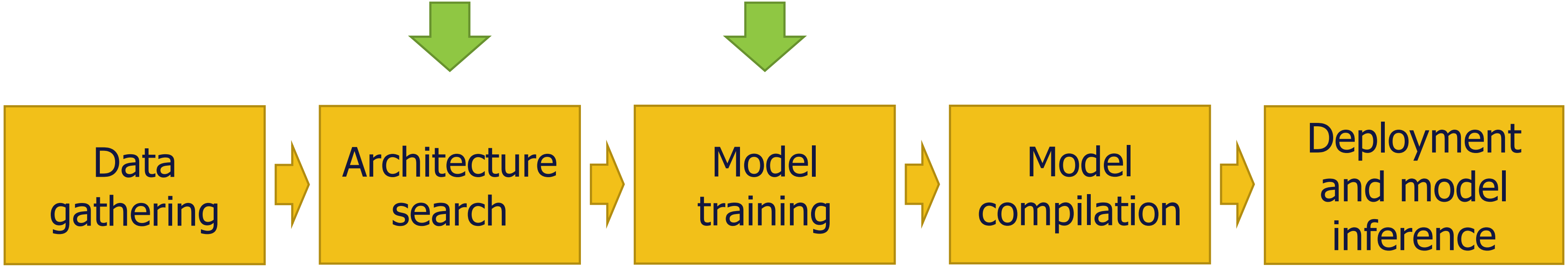
From https://www.tensorflow.org/model_optimization/guide/clustering

Knowledge distillation

INSTRUMENTAL

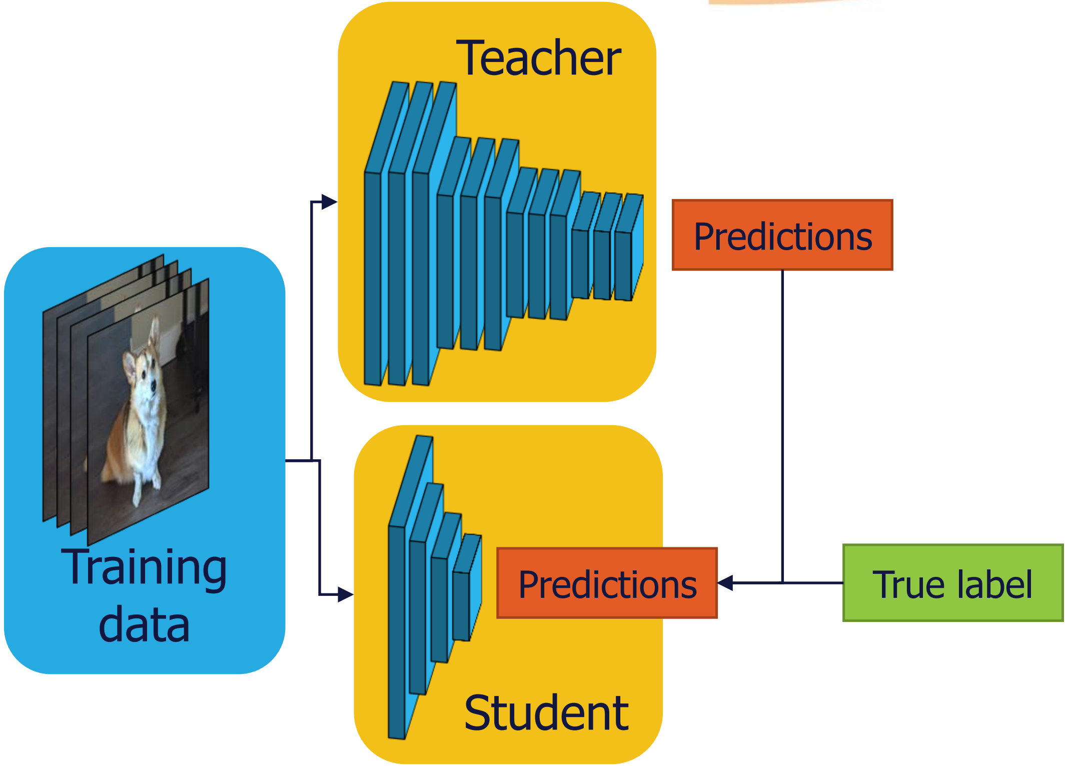
Knowledge distillation

- Smaller, more efficient "student" model learns to mimic the behavior of a larger, pre-trained "teacher" model



Knowledge distillation - example

- Only student model is trained
- Student model and teacher model run on the same images
- Error is propagated back for student mode



Knowledge distillation - example

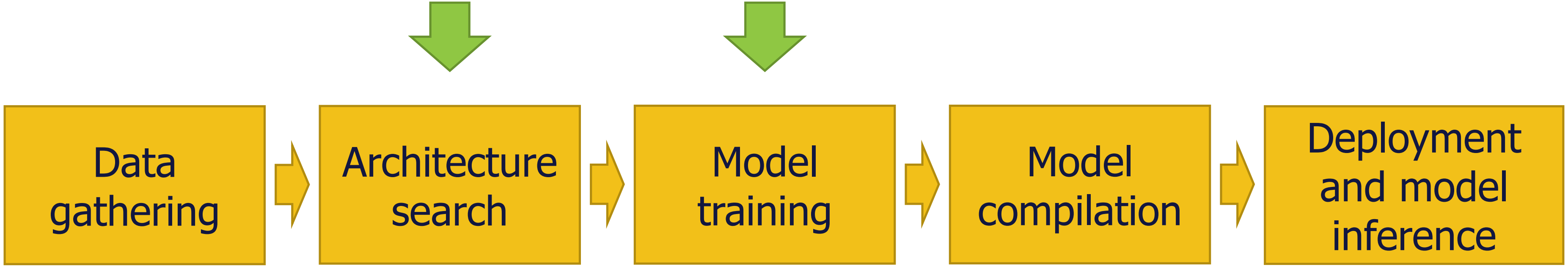
- Image classification example

	Base model	No. of parameters	Test accuracy
Teacher model	VGG16	27 M	77%
Student model with Distillation	VGG16 pruned	296 k	75%
Student model without Distillation	VGG16 pruned	296 k	64%

From https://www.analyticsvidhya.com/blog/2022/01/knowledge-distillation-theory-and-end-to-end-case-study/

Optimizing model architecture

- Finding smaller models which have less parameters and have faster predictions



Model architecture - example

- Object detection – YOLOv5 family

Model	Size, MB	mAP, COCO	CPU inf, ms	GPU (V100) inf, ms
YOLOv5n	4.1	45.7	45	6.3
YOLOv5s	14.8	56.8	98	6.4
YOLOv5m	42.8	64.1	224	8.2
YOLOv5l	93.6	67.3	430	10.1
YOLOv5x	174.1	68.9	766	12.1

From https://github.com/ultralytics/yolov5

Summary

Comparing optimization techniques

Optimization	Size decrease	Speed increase
Inference framework	Low	High
Pruning	High	Low
Quantization	High	High
Weight clustering	Low	Low
Architecture search	High	High

Recommendations on choosing model compression techniques

- Use a test dataset to assess performance changes
  - Impacts vary (e.g., less for image classification, more for object detection)
- Integrate compilation optimizations into training for optimal model selection
- Test on target hardware for deployment
- Balance trade-offs: Understand acceptable accuracy loss and business metric impact

Open-source vs commercial

- Open-source
  - Inference or training frameworks with built-in solutions
- Commercial
  - A number of companies specifically focus on optimizing your models (some of them are Alliance Members)

Open-source vs commercial

- Begin with an open-source solution for quick optimization gains
- Consider commercial for specialized hardware (e.g., mobile) with easy trial options
- Use a test set to validate accuracy trade-offs for both approaches

Conclusions

- Model compression is vital for cloud-to-edge ML pipelines
- Streamlined training pipeline enables easy exploration of approaches
- Integrating compression in training pipeline ensures optimal accuracy

Resources

- Inference framework guides
  - OpenVINO https://docs.openvino.ai/latest/openvino_docs_model_optimization_guide.html
  - TFLite https://www.tensorflow.org/lite/performance/model_optimization
  - ONNX https://onnxruntime.ai/docs/performance/model-optimizations/
- Books
  - TinyML https://www.oreilly.com/library/view/tinyml/9781492052036/
  - Deep Learning with PyTorch https://livebook.manning.com/book/deep-learning-with-pytorch/