Lecture 06:
Distillation, Low Rank Decomposition, Neural Architecture Search and Dynamic Computing

Notes

- Lab 1 due Mar 1st at 11:59 PM.
- Lab 2 will be post this Sunday night.

Format

- Due on March 20th 11:59pm (1 page)
  - Name + NetID of each group member
  - Project summary (1 paragraph)
  - Project plan (1 paragraph, gantt chart (optional))
  - Individual responsibilities (1 paragraph)
- You should start forming teams of **2–3 students**, which is the **strongly recommended project size**.
- Mark breakdown:
  - Proposal (1 page) 5%
  - Final presentation 10%
  - Final report 10%

Format

- Goal: Explore new and creative ideas in efficient AI and system.
- Your project should be feasible and well-scoped. Example topics include (but are not limited to):
  - Efficiency techniques on Application-specific Neural Network (Efficient AI + X)
  - Efficient large-model prefilling and decoding for faster execution
  - CNN/Transformer implementation on edge device
  - Efficient DNN training
  - Feel free to brainstorm your own ideas!
  - ….

Sample Projects

- Optimizing Communication in Memory-constrained 3DGS Training (Efficient AI + Graphics)
- Efficient Visual Autoregressive Modelling via Model Compression (LLM pruning)
- Multi-agent LLM Debates (LLM system)
- EfficentMCUNet-Based Face Detection on Raspberry Pi (AI system)
- Efficiency AI for Drug Delivery (AI + Science)

Recap

- Basic Data Formats
  - Fixed point (INT)
  - Floating point (FP)
  - Block floating point (BFP)
- Quantization methods
  - Taxonomy of Quantization
  - Learnable adaptive quantization scheme
  - Quantization for LLM

Topics

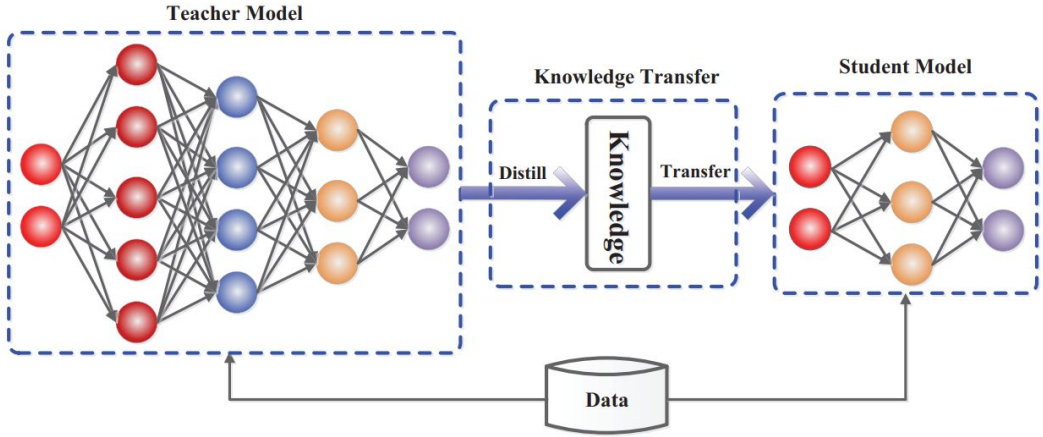
- Distillation
- Neural architecture search (NAS)
- Low-rank factorization
- Dynamic / Conditional Computing

Topics

- Distillation
- Neural architecture search (NAS)
- Low-rank factorization
- Dynamic / Conditional Computing

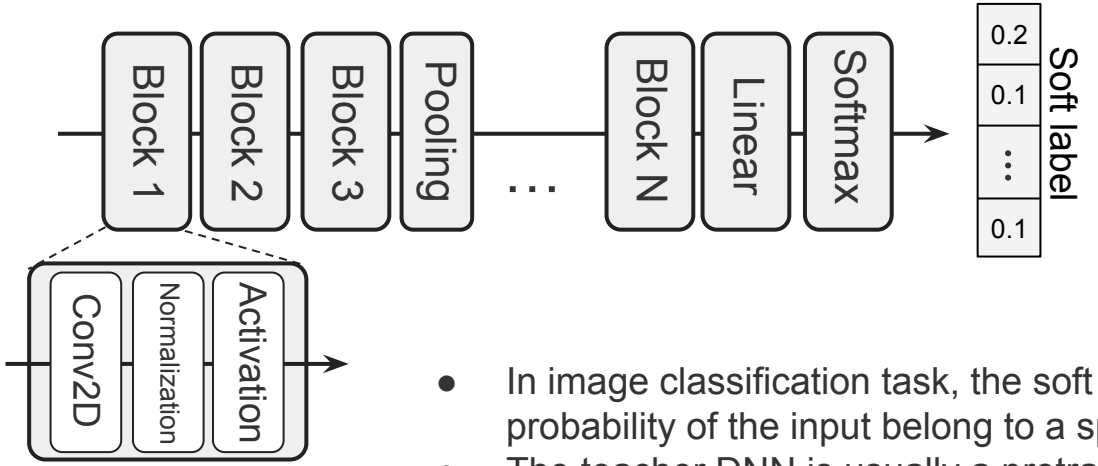
Knowledge Distillation Basics

- Transferring knowledge from a large and complex model or set of models to a single, smaller model that can be effectively deployed in real-world scenarios with practical limitations.



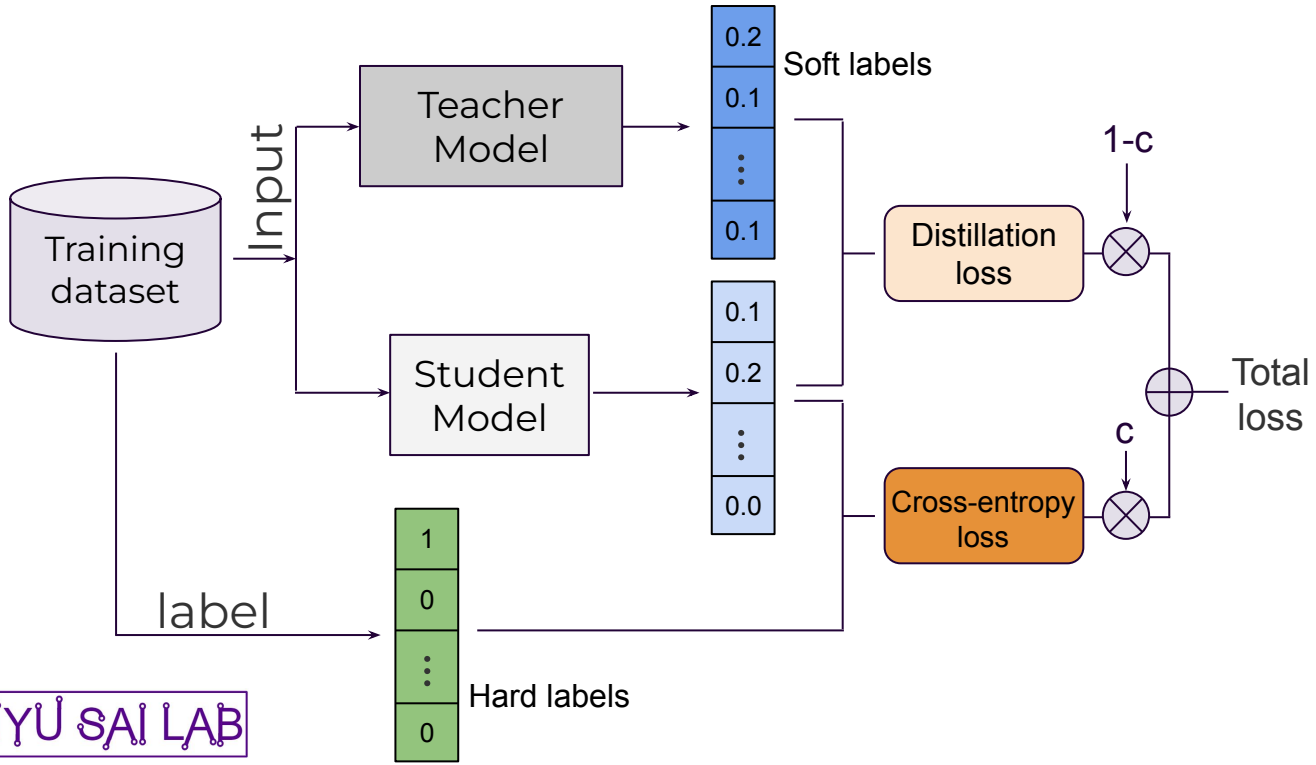
Hinton, Geoffrey, Oriol Vinyals, and Jeff Dean. "Distilling the knowledge in a neural network." *arXiv preprint arXiv:1503.02531* (2015).

Knowledge Distillation Basics



- In image classification task, the soft labels indicate the probability of the input belong to a specific class.
- The teacher DNN is usually a pretrained model, which has a much larger size than the student model.
- The student model is smaller than the teacher model and does not need to have the same architecture as the teacher.

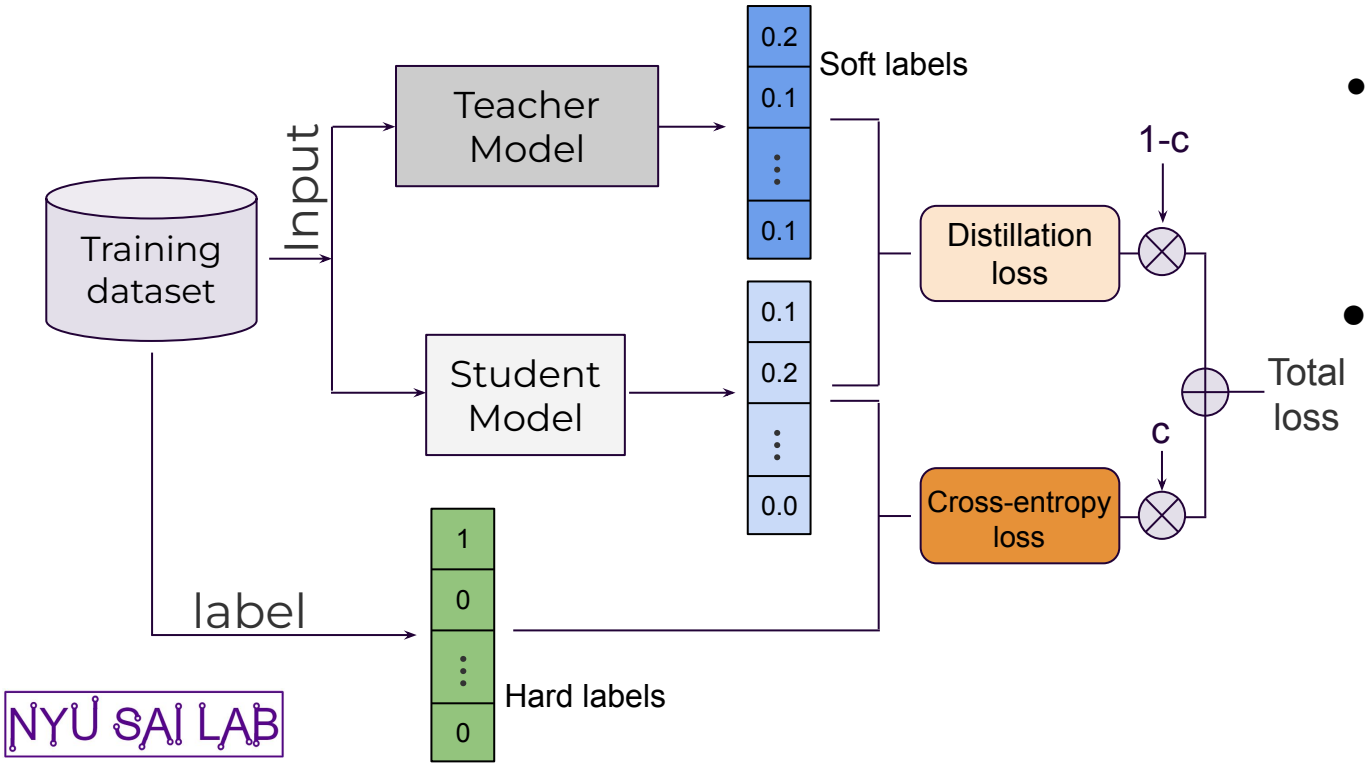
Response-Based Knowledge Distillation



- During the training process, only the weights within the student model got updated.
- The teacher model can be either pretrained or trained together with student.
- c tends to be close to 1: ~0.9.

NYU SAI LAB

Response-Based Knowledge Distillation



- Soft labels contains more information than the hard label, which will better guide the student model to learn.
- This soft supervision reveals class relationships hidden in hard labels, improving the student's generalization and enabling **model compression** without major accuracy loss.

NYU SAI LAB

KL Divergence

- In mathematical statistics, the Kullback–Leibler (KL) divergence, denoted $D_{KL}(P \,//\, Q)$, is a measure of how much a model probability distribution Q is different from a true probability distribution P, where both P and Q are discrete.

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x)\ \log\left(\frac{P(x)}{Q(x)}\right)$$

- If P(x) = Q(x) for all x, then the KL divergence equals 0.
- Otherwise, KL divergence is greater than 0.

Response-Based Knowledge Distillation

- The soft labels are computed as follows:

$$p(z_i, T) = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}$$

T is the temperature
zi is called logits
p is the soft label

- Kullback Leibler (KL) divergence is usually used to generate the distillation loss.

$$L(y_{true}, y_{pred}) = y_{\text{true}} \cdot \log \frac{y_{\text{true}}}{y_{\text{pred}}} = y_{\text{true}} \cdot (\log y_{\text{true}} - \log y_{\text{pred}})$$

- The loss function of the student model includes both distillation loss and the conventional cross-entropy loss.

NYU SAI LAB

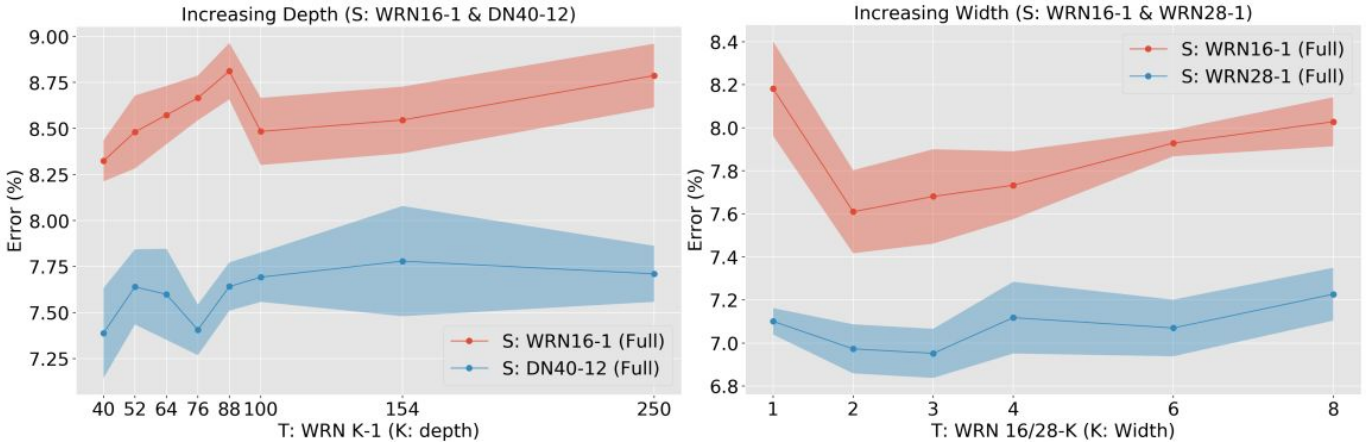
How Smart the Teacher Need to Be?

- Larger teachers, though they are more accurate by themselves, do not necessarily make for better teachers. A too large teacher model can even harm the performance of the student.

- As the teacher grows in capacity and accuracy, the student often finds it difficult to emulate the teacher.

Cho, Jang Hyun, and Bharath Hariharan. "On the efficacy of knowledge distillation." *Proceedings of the IEEE/CVF international conference on computer vision*. 2019.

How Smart the Teacher Need to Be?

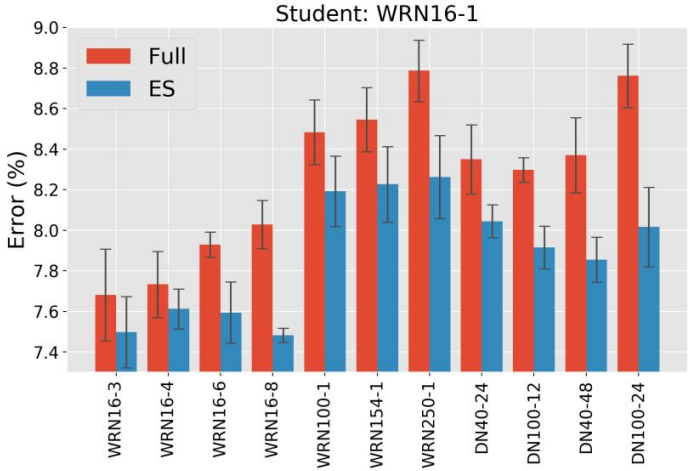


- Increasing teacher capacity (depth: left, width: right) and thus accuracy does not necessarily increase the accuracy of the student network.
- If the teacher network is too large, the student is unable to find a solution in its space that corresponds well to the largest teacher.

NYU SAI LAB

Cho, Jang Hyun, and Bharath Hariharan. "On the efficacy of knowledge distillation." *Proceedings of the IEEE/CVF international conference on computer vision*. 2019.

How Smart the Teacher Need to Be?

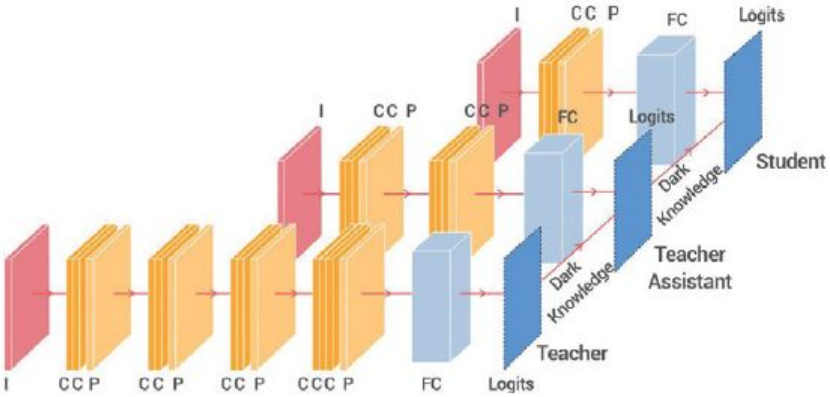


- We may perform grid-search to find the optimal teacher network architecture, but that is too expensive.
- In particular, we propose to early stop (ES) the training process.

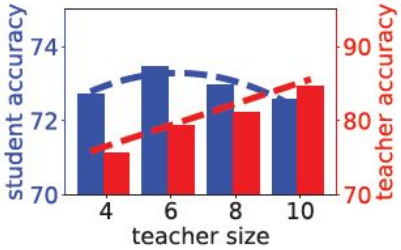
NYU SAI LAB

Cho, Jang Hyun, and Bharath Hariharan. "On the efficacy of knowledge distillation." *Proceedings of the IEEE/CVF international conference on computer vision*. 2019.

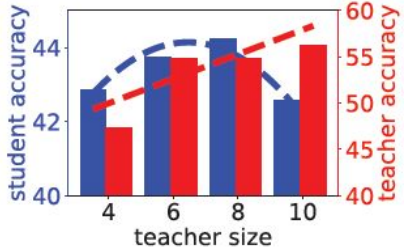
Improved Knowledge Distillation via Teacher Assistant



- Knowledge distillation is not always effective, especially when the gap (in size) between teacher and student is large.
- TA models are distilled from the teacher, and the student is then only distilled from the TAs



a) CIFAR-10



b) CIFAR-100

NYU SAI LAB

Mirzadeh, Seyed Iman, et al. "Improved knowledge distillation via teacher assistant." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. No. 04. 2020.

Improved Knowledge Distillation via Teacher Assistant

Table 1: Comparison on evaluation accuracy between our method (TAKD) and baselines. For CIFAR, plain (S=2, TA=4, T=10) and for ResNet (S=8, TA=20, T=110) are used. For ImageNet, ResNet (S=14, TA=20, T=50) is used. Higher numbers are better.

Model	Dataset	NOKD	BLKD	TAKD
CNN	CIFAR-10	70.16	72.57	73.51
	CIFAR-100	41.09	44.57	44.92
ResNet	CIFAR-10	88.52	88.65	88.98
	CIFAR-100	61.37	61.41	61.82
ResNet	ImageNet	65.20	66.60	67.36

- In this paper, the authors propose to use intermediate-size networks to fill in the gap between them.
- First, the TA network is distilled from the teacher. Then, the TA plays the role of a teacher and trains the student via distillation.

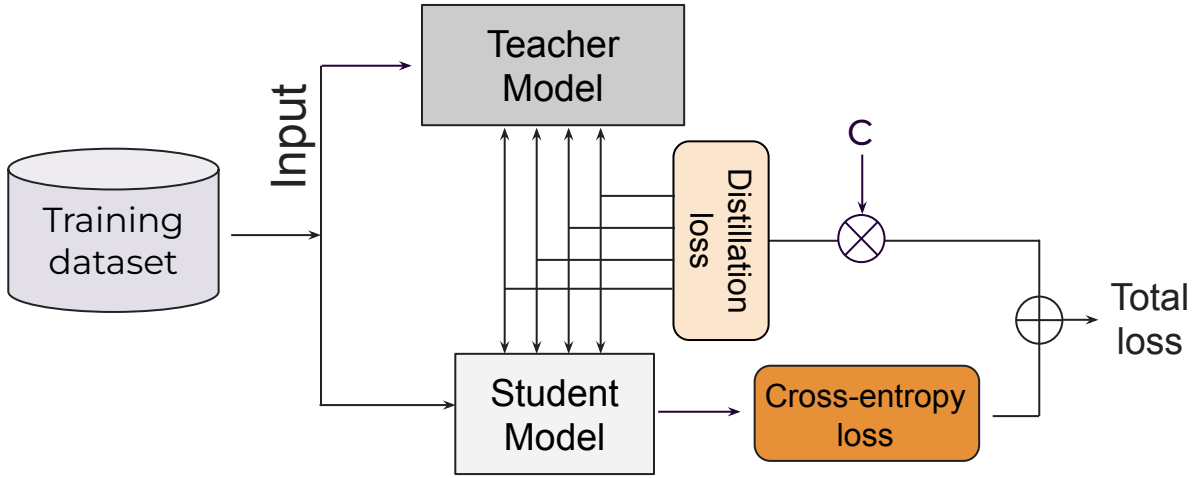
Mirzadeh, Seyed Iman, et al. "Improved knowledge distillation via teacher assistant." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. No. 04. 2020.

Feature-Based Knowledge Distillation

- Response-based knowledge usually relies on the output of the last layer, e.g., soft targets, and thus fails to address the intermediate-level supervision from the teacher model.
- Both the output of the last layer and the output of intermediate layers, i.e., feature maps, can be used as the knowledge to supervise the training of the student model.
- It transfers intermediate representations, allowing the student to learn how the teacher processes information more deeply, resulting in improved generalization and faster convergence.

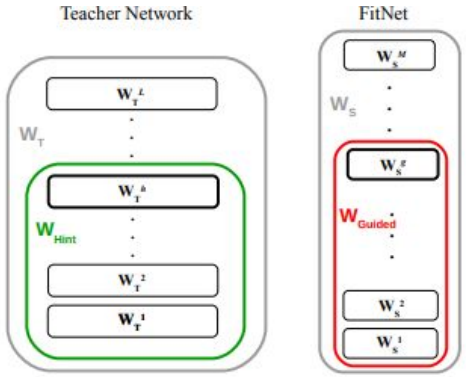
Srivastava, Rupesh K., Klaus Greff, and Jürgen Schmidhuber. "Training very deep networks." *Advances in neural information processing systems* 28 (2015).

Feature-Based Knowledge Distillation

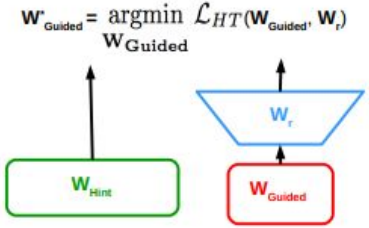


- For feature based Knowledge, distillation, the intermediate results from the teacher model is extracted and guide the training for the student DNN.

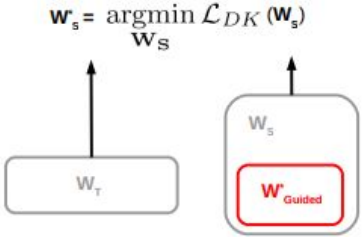
FitNets: Hints for Thin Deep Nets



(a) Teacher and Student Networks



(b) Hints Training



(c) Knowledge Distillation

$$\mathcal{L}_{HT}(\mathbf{W_{Guided}}, \mathbf{W_r}) = \frac{1}{2}||u_h(\mathbf{x}; \mathbf{W_{Hint}}) - r(v_g(\mathbf{x}; \mathbf{W_{Guided}}); \mathbf{W_r})||^2$$

HINT-BASED TRAINING

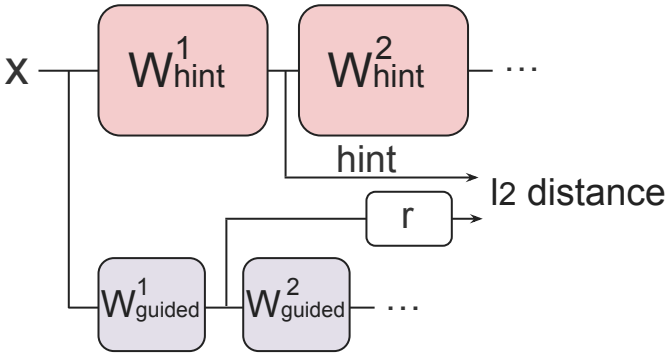
NYU SAI LAB

Romero, Adriana, et al. "Fitnets: Hints for thin deep nets." *arXiv preprint arXiv:1412.6550* (2014).

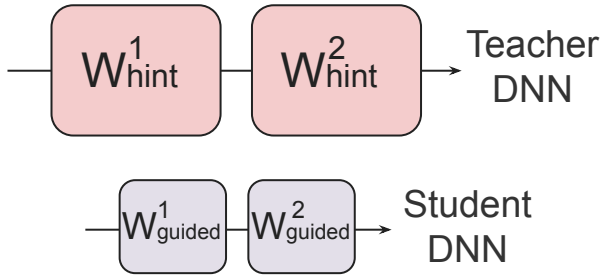
FitNets: Hints for Thin Deep Nets

$$\mathcal{L}_{HT}(\mathbf{W_{Guided}}, \mathbf{W_r}) = \frac{1}{2}||u_h(\mathbf{x}; \mathbf{W_{Hint}}) - r(v_g(\mathbf{x}; \mathbf{W_{Guided}}); \mathbf{W_r})||^2$$

HINT-BASED TRAINING

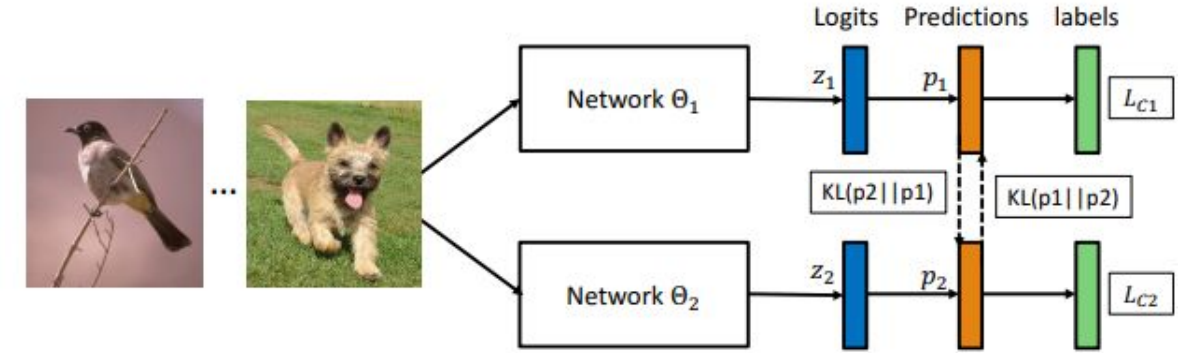


- r(.) consists of conv layers, in order to produce the outputs with the same dimension as the teacher DNN.



Romero, Adriana, et al. "Fitnets: Hints for thin deep nets." *arXiv preprint arXiv:1412.6550* (2014).

Distillation Schemes: Online Distillation



- An ensemble of students learn collaboratively and teach each other throughout the training process
- A similar method can be extended to a larger group of students.

NYU SAI LAB

Zhang, Ying, et al. "Deep mutual learning." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

Distillation Schemes: Online Distillation

Algorithm 1: Deep Mutual Learning

Input: Training set $\mathcal{X}$, label set $\mathcal{Y}$, learning rate $\gamma_{1,t}$ and $\gamma_{2,t}$.
Initialize: Models Θ_1 and Θ_2 to different initial conditions.
Repeat :
$t = t + 1$
Randomly sample data $\boldsymbol{x}$ from $\mathcal{X}$.
1: Update the predictions $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ of $\boldsymbol{x}$ by (1) for the current mini-batch
2: Compute the stochastic gradient and update Θ_1 :

$$\Theta_1 \leftarrow \Theta_1 + \gamma_{1,t} \frac{\partial L_{\Theta_1}}{\partial \Theta_1}$$

3: Update the predictions $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ of $\boldsymbol{x}$ by (1) for the current mini-batch
4: Compute the stochastic gradient and update Θ_2 :

$$\Theta_2 \leftarrow \Theta_2 + \gamma_{2,t} \frac{\partial L_{\Theta_2}}{\partial \Theta_2}$$

Until : convergence

$$D_{KL}(\boldsymbol{p}_2 \| \boldsymbol{p}_1) = \sum_{i=1}^{N} \sum_{m=1}^{M} p_2^m(\boldsymbol{x}_i) \log \frac{p_2^m(\boldsymbol{x}_i)}{p_1^m(\boldsymbol{x}_i)}$$

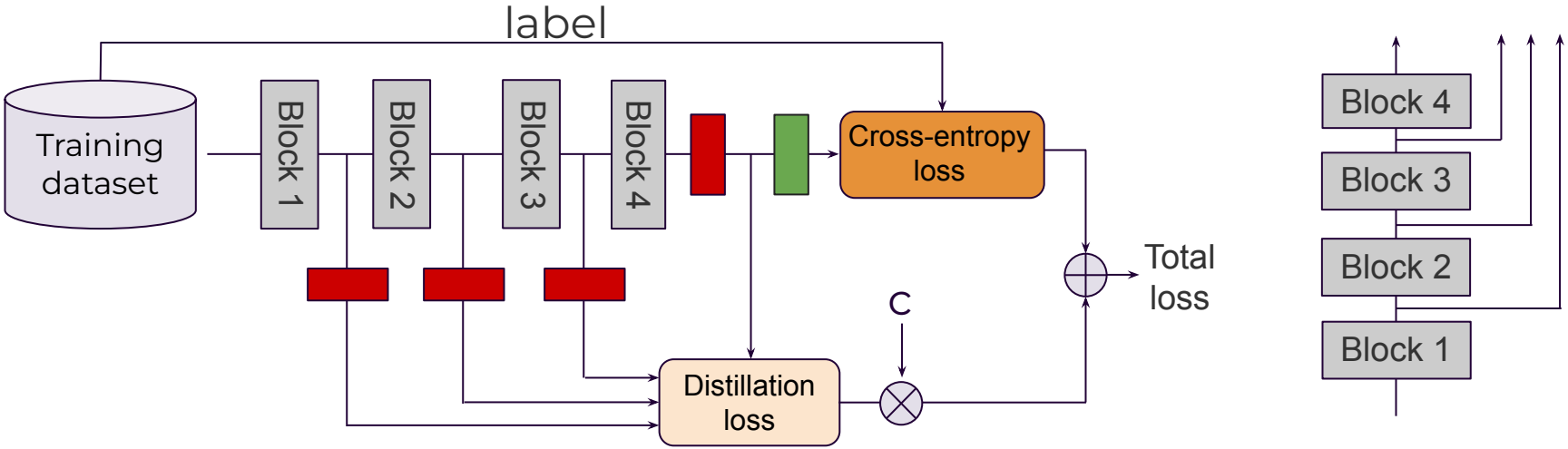
$$L_{\Theta_1} = L_{C_1} + D_{KL}(\boldsymbol{p}_2 \| \boldsymbol{p}_1)$$

$$L_{\Theta_2} = L_{C_2} + D_{KL}(\boldsymbol{p}_1 \| \boldsymbol{p}_2)$$

- The same concept can be applied to situations involving multiple students.
- A cohort of students to learn mutually.

Zhang, Ying, et al. "Deep mutual learning." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

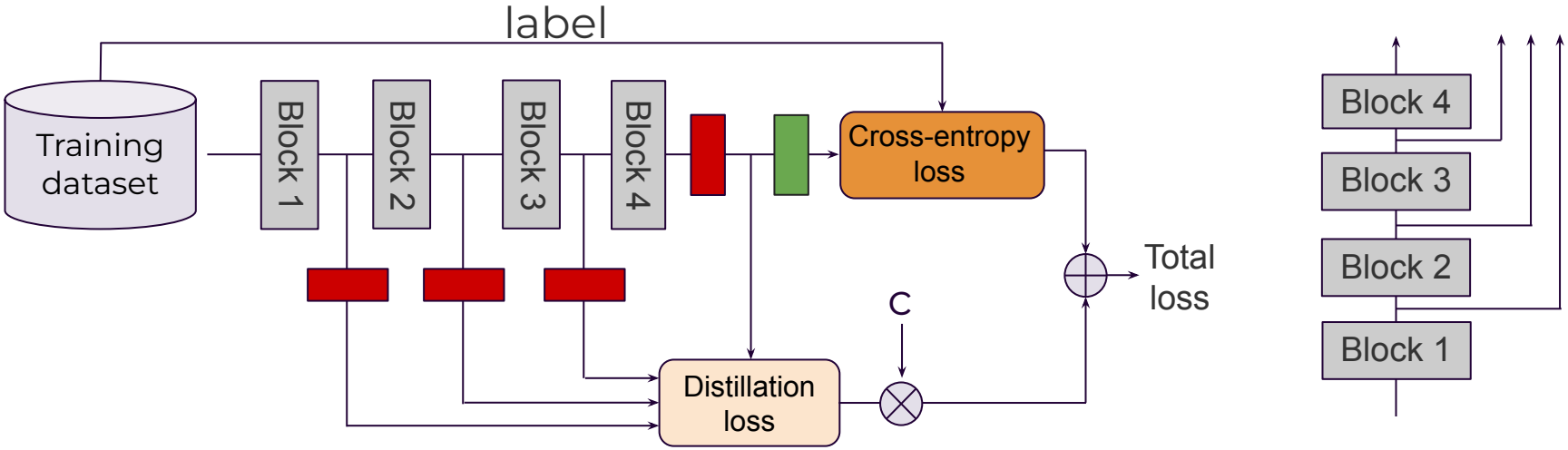
Self Distillation



- The networks are firstly divided into several components, and the knowledge from the deeper parts is condensed into the shallower ones.

Zhang, Linfeng, et al. "Be your own teacher: Improve the performance of convolutional neural networks via self distillation." *Proceedings of the IEEE/CVF international conference on computer vision*. 2019.

Self Distillation



- Self distillation works because a model can teach itself using its own softened outputs as guidance, encouraging smoother decision boundaries and better internal consistency.
- This process helps the network refine its representations, reduce overfitting, and extract richer knowledge from its own predictions without needing a separate teacher model.

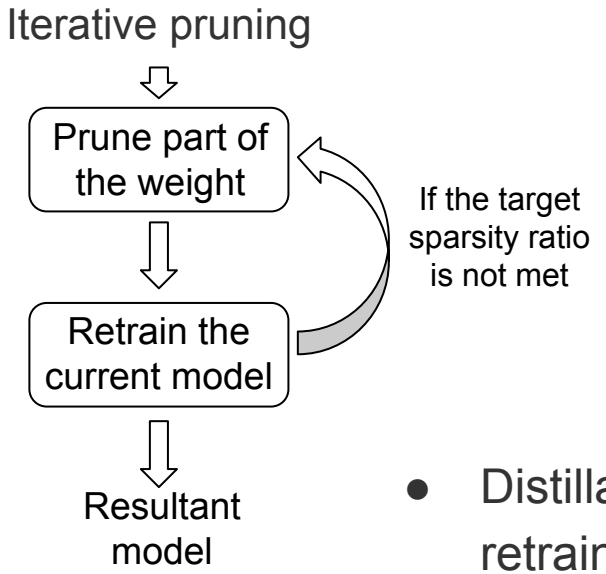
NYU SAI LAB

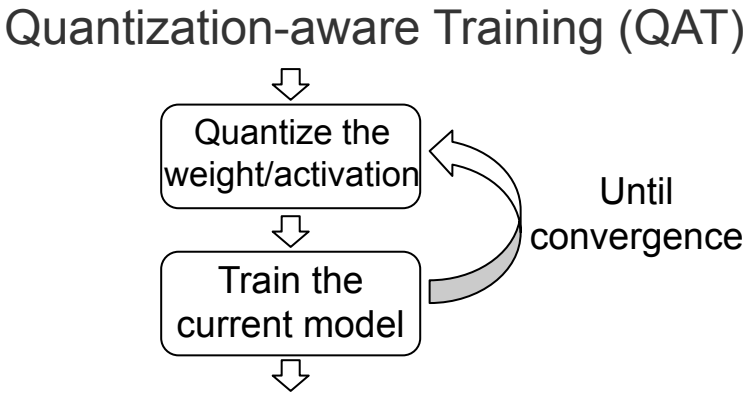
Zhang, Linfeng, et al. "Be your own teacher: Improve the performance of convolutional neural networks via self distillation." *Proceedings of the IEEE/CVF international conference on computer vision*. 2019.

Relation between Student and Teacher

- The student model should have some architectural similarities with the teacher.
  - It can be a mini version of teacher model (shallower, thinner)
  - It can be a pruned version of teacher
  - It can be a quantized version of teacher

Distillation for Quantization and Pruning

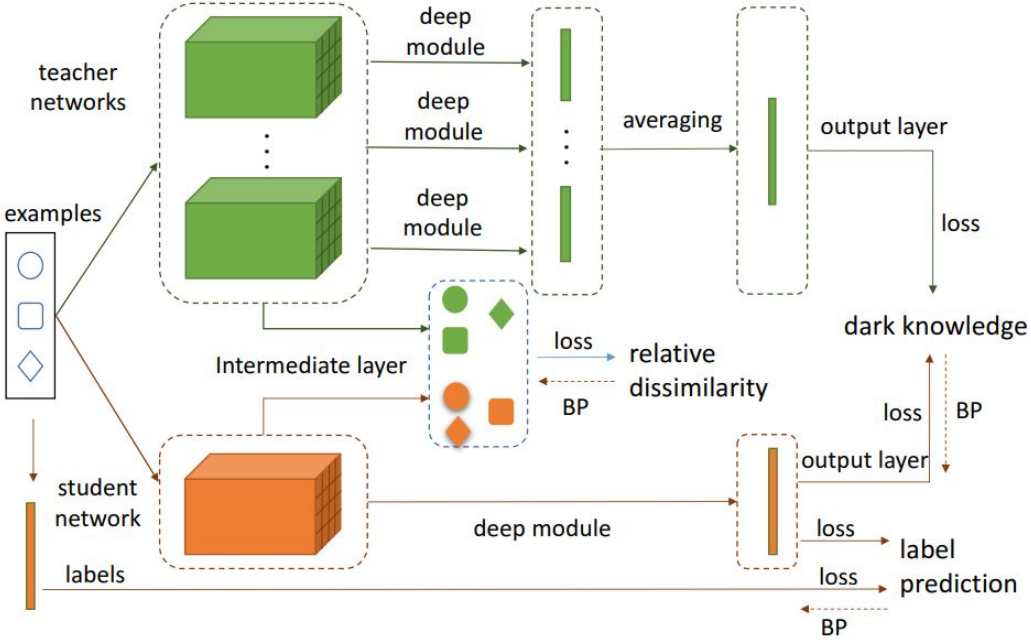




- Distillation can help streamline the finetuning or retraining process during iterative pruning or quantization-aware training.

NYU SAI LAB

Multi-teacher Distillation

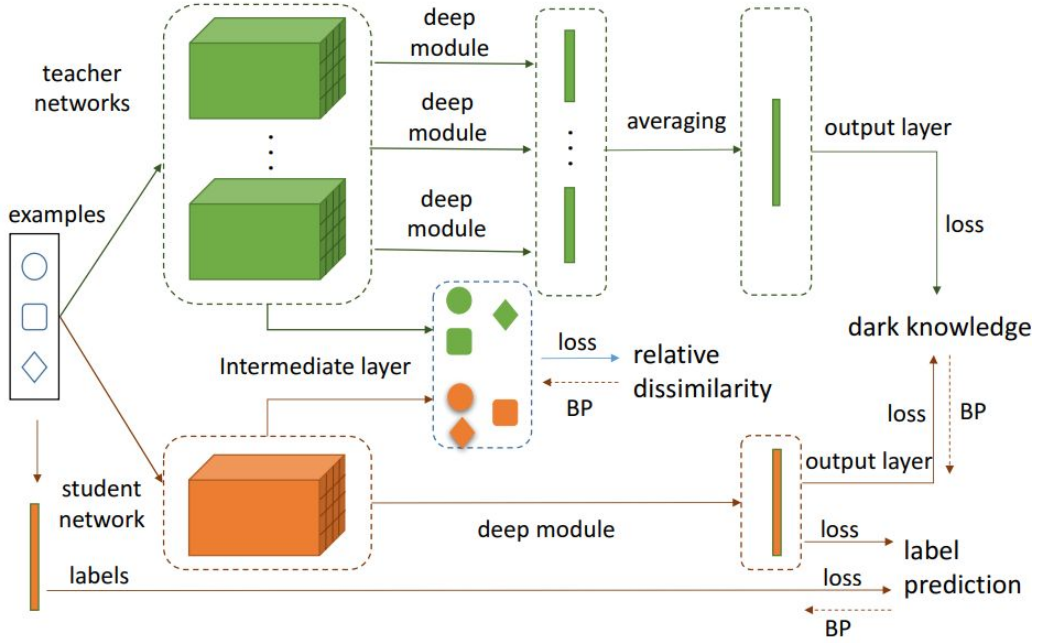


- We present a method to train a student network by incorporating multiple teacher networks not only in output layer by averaging the softened outputs (dark knowledge) from different networks, but also in the intermediate layers by imposing a constraint about the dissimilarity among examples.

NYU SAI LAB

You, Shan, et al. "Learning from multiple teacher networks." *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 2017.

Multi-teacher Distillation

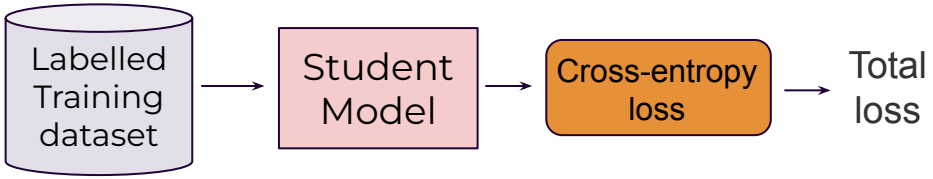


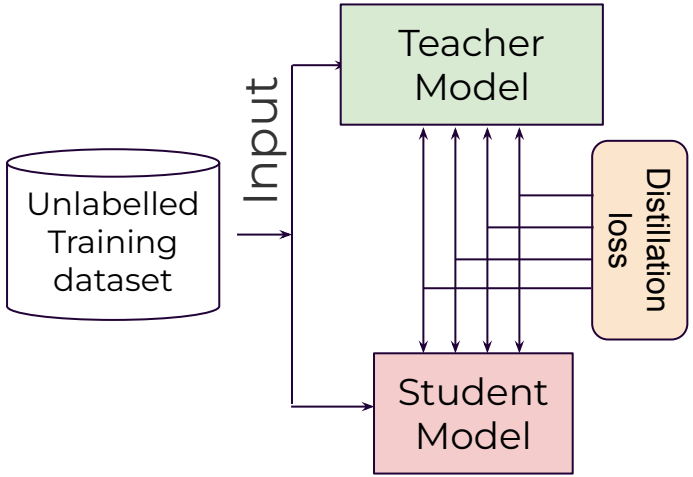
- Multiple teacher can better guide the student DNN to learn.
- Each teacher network can with different architecture, or even same architecture but different weights.

NYU SAI LAB

You, Shan, et al. "Learning from multiple teacher networks." *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 2017.

Cross-Modal Distillation

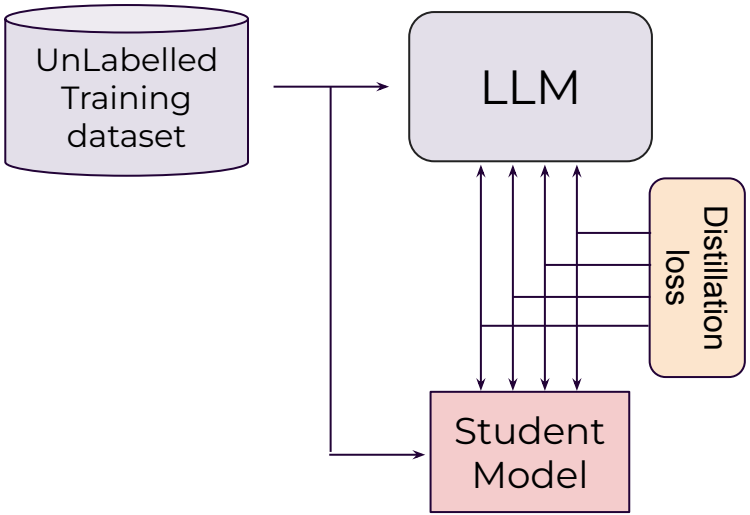




- If the labeled data on the new tasks is limited (e.g., segmentation), we can pass the unlabelled data to the teacher model pretrained for another similar task (e.g., classification), and use the feature-based knowledge distillation to better guide the student.
- The teacher network, despite being trained on different vision tasks, still demonstrates strong proficiency in understanding visual inputs.

NYU SAI LAB

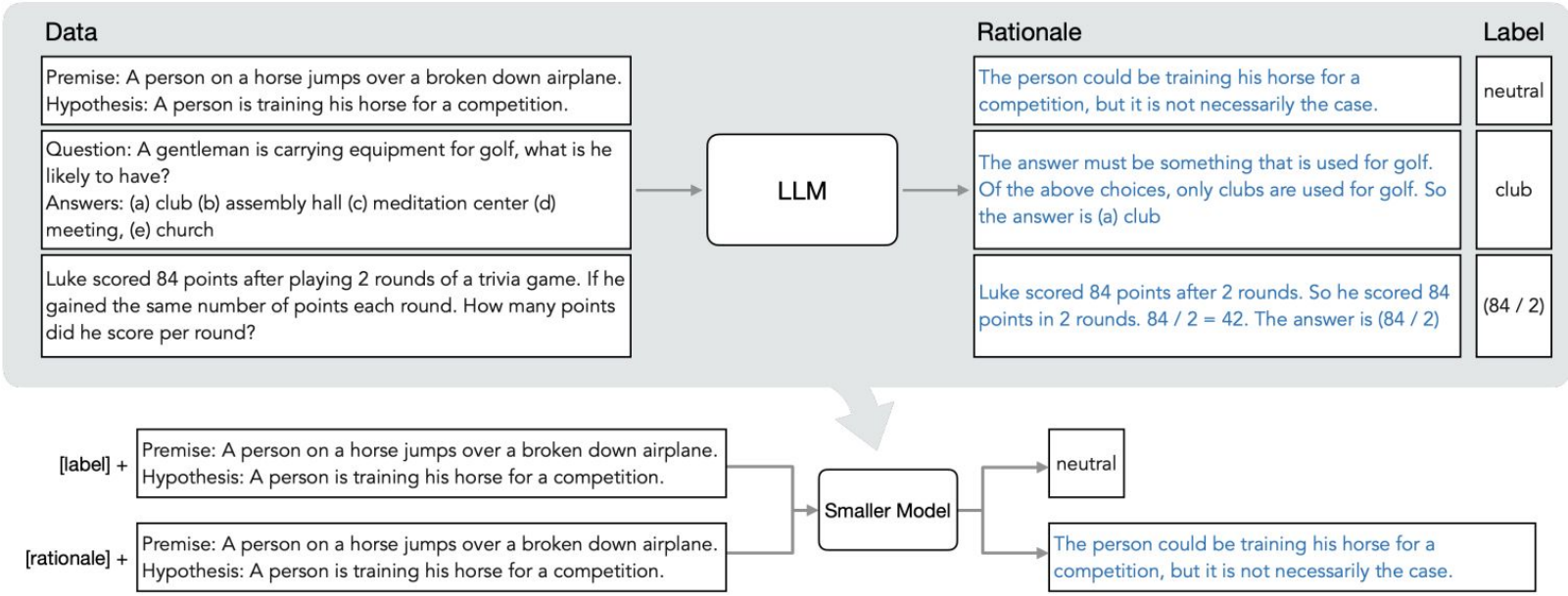
Distillation for Language Models



- Given the unlabelled data, LLM can leverage its knowledge to provide:
  - ground-truth label
  - Features (rationale)
- Distilling step-by-step is a new simple mechanism for training smaller models with less training data.
- This mechanism reduces the amount of training data required for both finetuning and distillation of LLMs into smaller model sizes.

Hsieh, Cheng-Yu, et al. "Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes." *arXiv preprint arXiv:2305.02301* (2023).

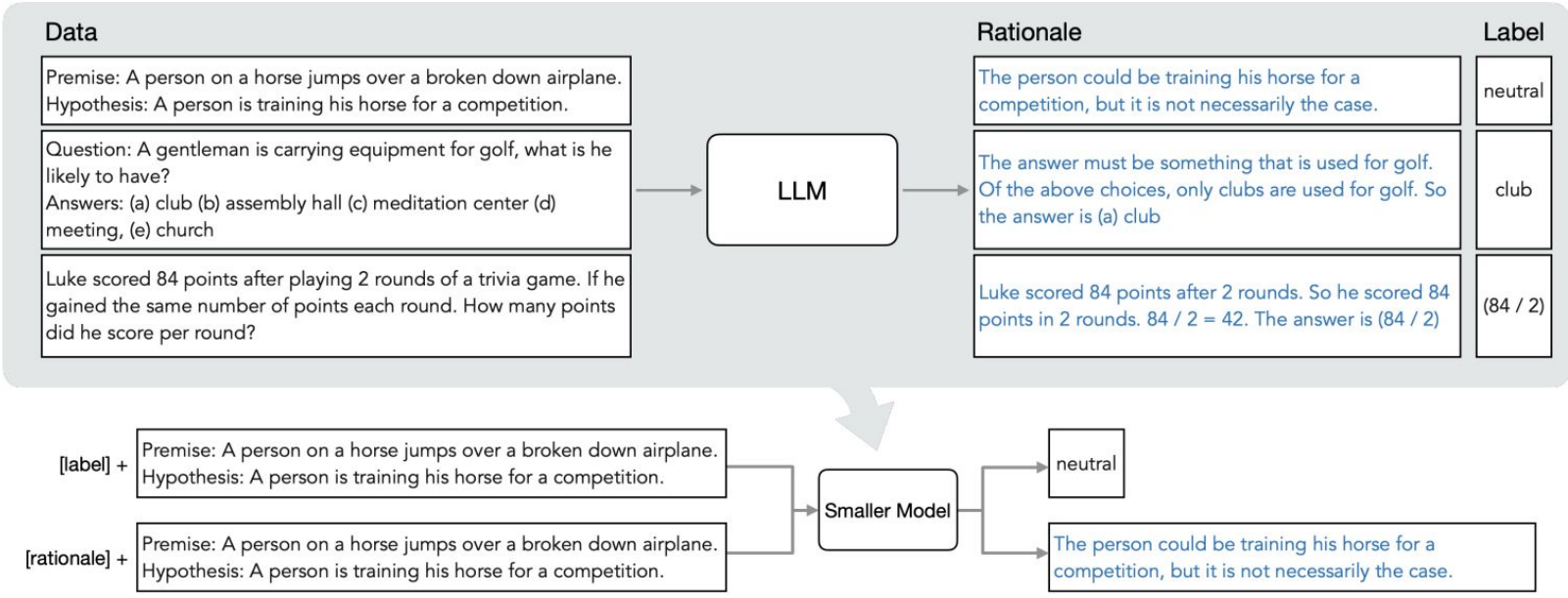
Distilling Step-by Step



- First, given an LLM and an unlabeled dataset, we prompt the LLM to generate output labels along with rationales to justify the labels

NYU SAI LAB

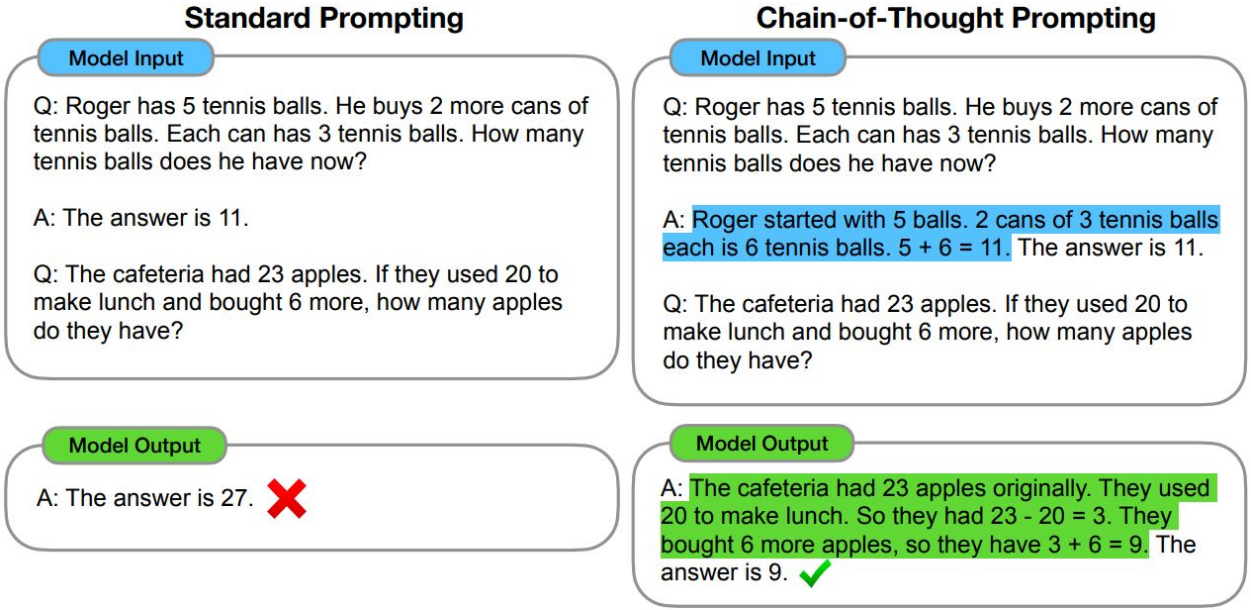
Distilling Step-by Step



- Second, we leverage these rationales in addition to the task labels to train smaller downstream models.

NYU SAI LAB

Chain-of-Thought (CoT)

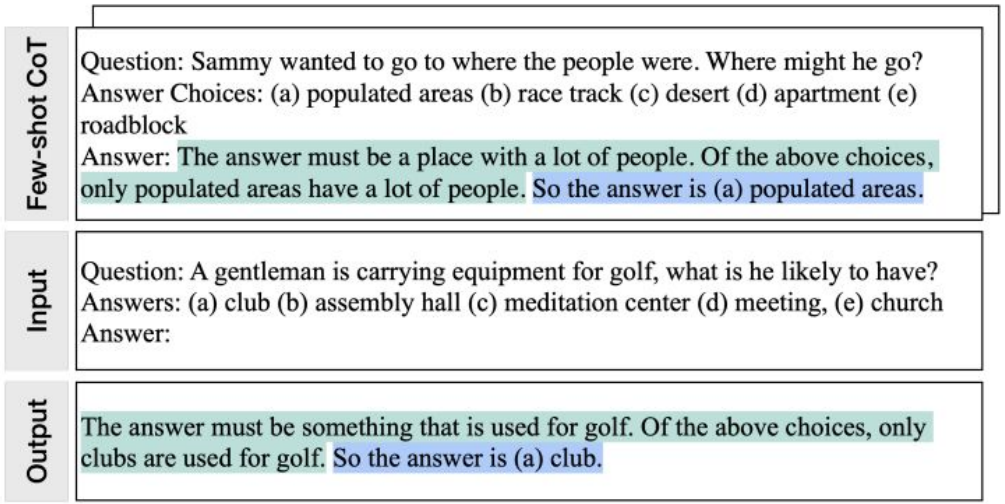


- CoT allows models to decompose multi-step problems into intermediate steps, which means that additional computation can be allocated to problems that require more reasoning steps.

NYU SAI LAB

Wei, Jason, et al. "Chain-of-thought prompting elicits reasoning in large language models." *Advances in neural information processing systems* 35 (2022): 24824-24837.

Distilling Step-by-Step



- We use few-shot Chain-of-thought prompting that contains both an example rationale (highlighted in green) and a label (highlighted in blue) to elicit rationales from an LLM on new input examples.
- We first design a prompt template p that articulates how the task should be solved.
- The rationale generation loss enables the model to learn to generate the intermediate reasoning steps for the prediction, and could therefore guide the model in better predicting the resultant label.

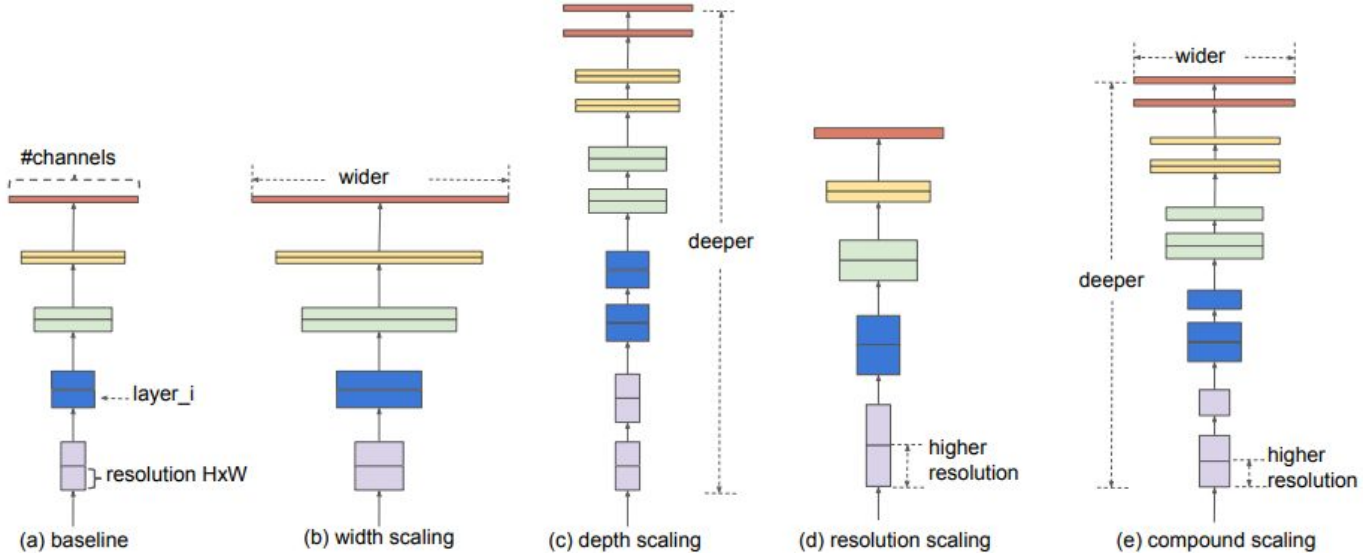
$$\mathcal{L} = \mathcal{L}_{\text{label}} + \lambda \mathcal{L}_{\text{rationale}}$$

NYU SAI LAB

Topics

- Distillation
- Neural architecture search (NAS)
- Low-rank factorization
- Dynamic / Conditional Computing

Neural Architecture Search (NAS)

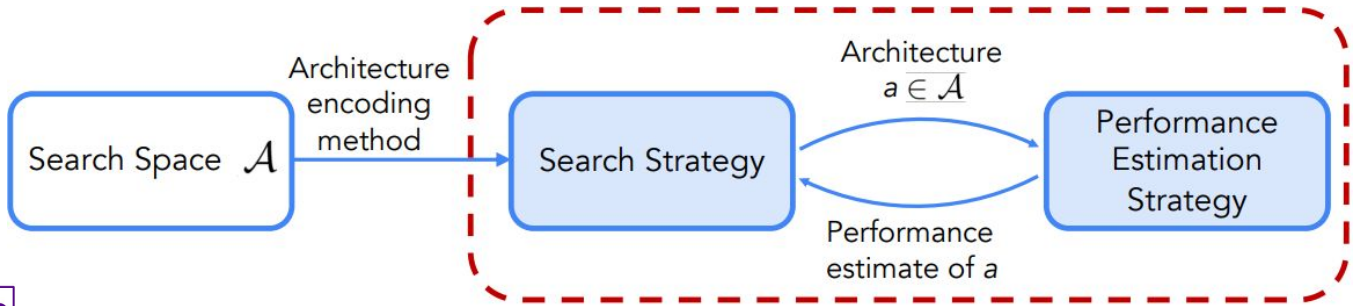


- How can a DNN be designed with an optimal architecture to maximize accuracy on the test dataset?

NYU SAI LAB

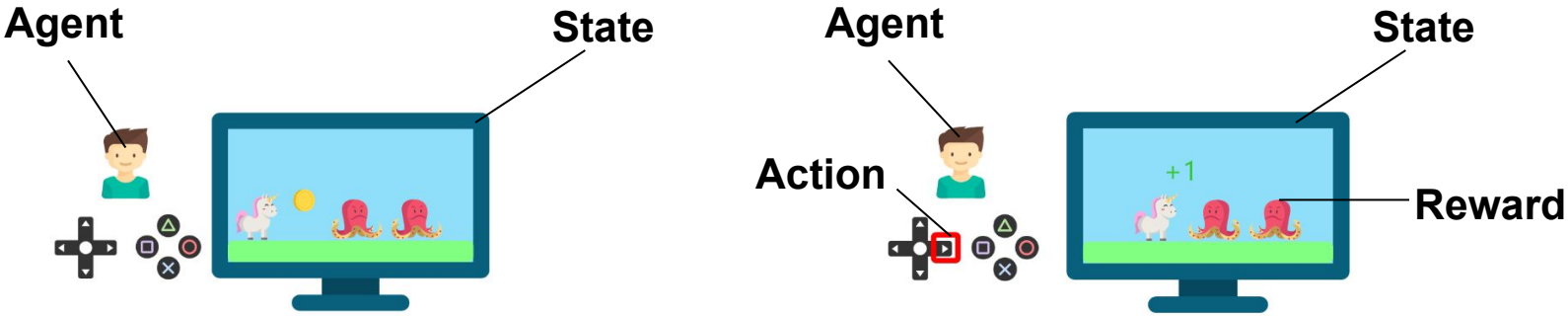
Neural Architecture Search (NAS)

- Neural Architecture Search (NAS) is an automated process for designing neural network architectures.
- Traditional methods of creating neural networks rely heavily on human expertise and manual experimentation, which can be time-consuming and suboptimal.
- NAS aims to alleviate these challenges by using algorithms to explore the vast space of possible network architectures to find the most efficient and effective designs for specific tasks.



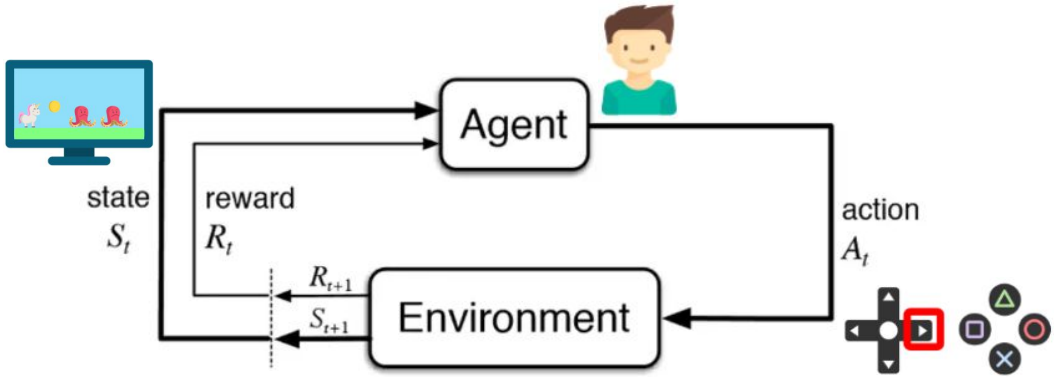
NYU SAI LAB

Deep Reinforcement Learning



- Reinforcement learning is an area of machine learning where an agent learns to make decisions by performing actions in an environment to maximize some cumulative reward.

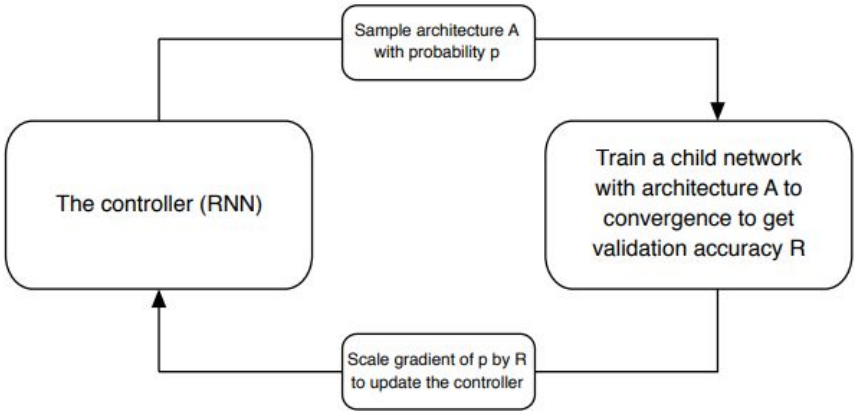
Deep Reinforcement Learning



- Agent: player
- State: game image
- Action: one button in gamepad
- Environment: game software

NYU SAI LAB

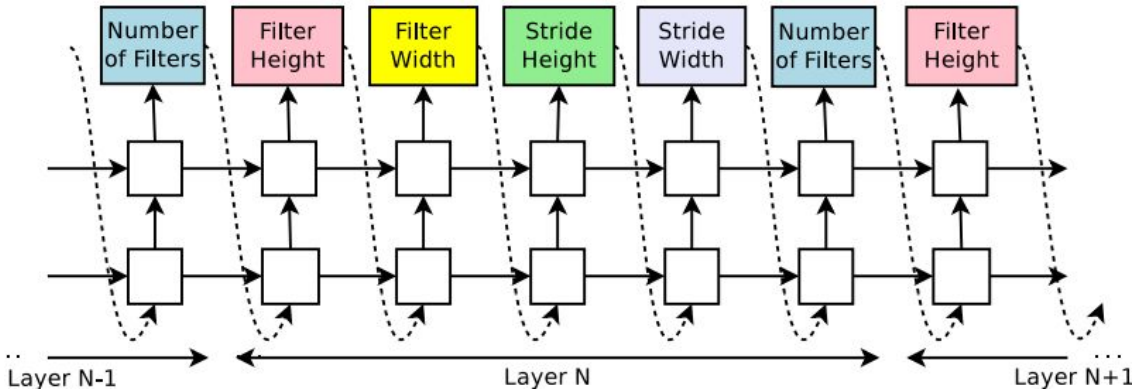
Learning Transferable Architectures for Scalable Image Recognition



- We use a recurrent network to generate the model descriptions of neural networks and train this RNN with reinforcement learning to maximize the expected accuracy of the generated architectures on a validation set.

Zoph, Barret, and Quoc V. Le. "Neural architecture search with reinforcement learning." *arXiv preprint arXiv:1611.01578* (2016).

Learning Transferable Architectures for Scalable Image Recognition



- Reward is based on the accuracy on the validation dataset.
- The action of the current step is based on the all the previous history on action is has taken.

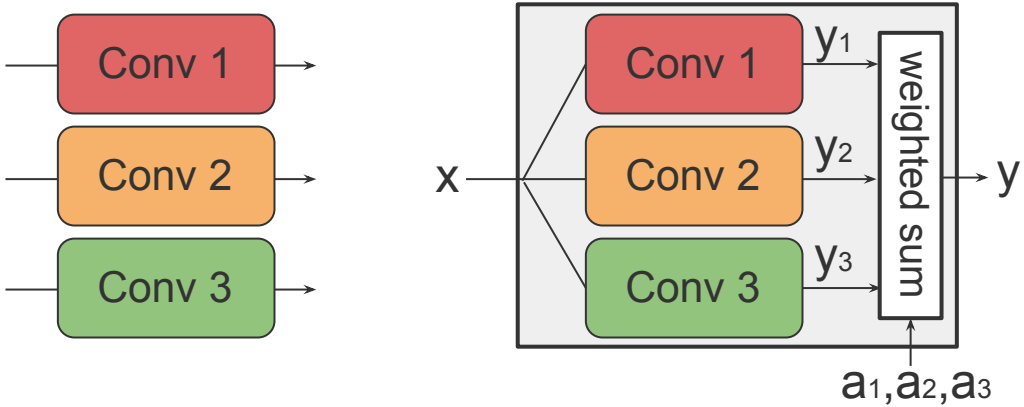
Zoph, Barret, and Quoc V. Le. "Neural architecture search with reinforcement learning." *arXiv preprint arXiv:1611.01578* (2016).

Learning Transferable Architectures for Scalable Image Recognition

model	depth	# params	error rate (%)
DenseNet ($L = 40, k = 12$) [26]	40	1.0M	5.24
DenseNet($L = 100, k = 12$) [26]	100	7.0M	4.10
DenseNet ($L = 100, k = 24$) [26]	100	27.2M	3.74
DenseNet-BC ($L = 100, k = 40$) [26]	190	25.6M	3.46
Shake-Shake 26 2x32d [18]	26	2.9M	3.55
Shake-Shake 26 2x96d [18]	26	26.2M	2.86
Shake-Shake 26 2x96d + cutout [12]	26	26.2M	2.56
NAS v3 [71]	39	7.1M	4.47
NAS v3 [71]	39	37.4M	3.65
NASNet-A (6 @ 768)	-	3.3M	3.41
NASNet-A (6 @ 768) + cutout	-	3.3M	2.65
NASNet-A (7 @ 2304)	-	27.6M	2.97
NASNet-A (7 @ 2304) + cutout	-	27.6M	2.40
NASNet-B (4 @ 1152)	-	2.6M	3.73
NASNet-C (4 @ 640)	-	3.1M	3.59

Zoph, Barret, et al. "Learning transferable architectures for scalable image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

DARTS: Differentiable Architecture Search



$y_1 = \text{Conv}(x, w_1)$

$y_2 = \text{Conv}(x, w_2)$

$y_3 = \text{Conv}(x, w_3)$

$$s_1 = \frac{e^{a_1}}{e^{a_1} + e^{a_2} + e^{a_3}}$$

$$s_2 = \frac{e^{a_2}}{e^{a_1} + e^{a_2} + e^{a_3}}$$

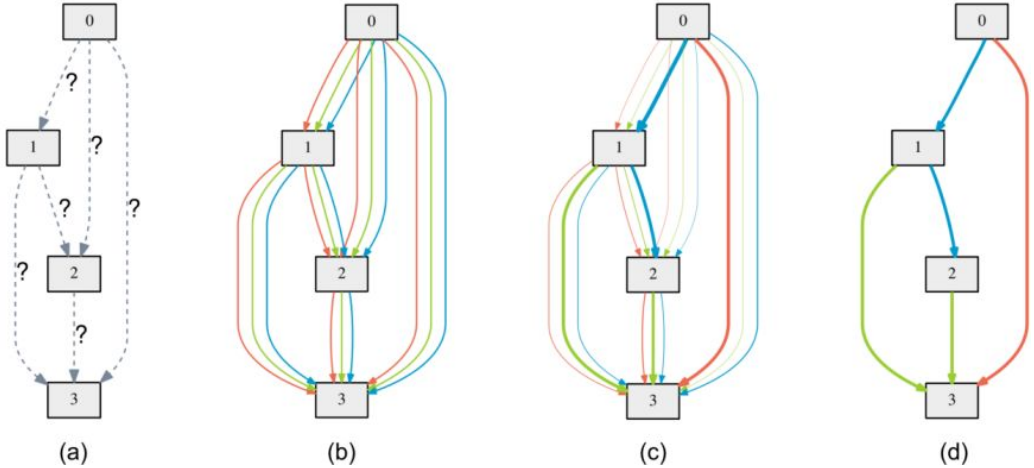
$$s_3 = \frac{e^{a_3}}{e^{a_1} + e^{a_2} + e^{a_3}}$$

$y = s_1 y_1 + s_2 y_2 + s_3 y_3$

- At the end of search, we simply select the conv layer with the highest s.
- We can also add additional regularization term to push s to be one-hot.

NYU SAI LAB

DARTS: Differentiable Architecture Search



$$\bar{o}^{(i,j)}(x) = \sum_{o \in \mathcal{O}} \frac{\exp(\alpha_o^{(i,j)})}{\sum_{o' \in \mathcal{O}} \exp(\alpha_{o'}^{(i,j)})} o(x)$$

$$\min_{\alpha} \quad \mathcal{L}_{val}(w^*(\alpha), \alpha)$$
$$\text{s.t.} \quad w^*(\alpha) = \text{argmin}_w \ \mathcal{L}_{train}(w, \alpha)$$

- DARTs continuous relax of the search space by placing a mixture of candidate operations on each edge, and combines the outputs from each branch using softmax.

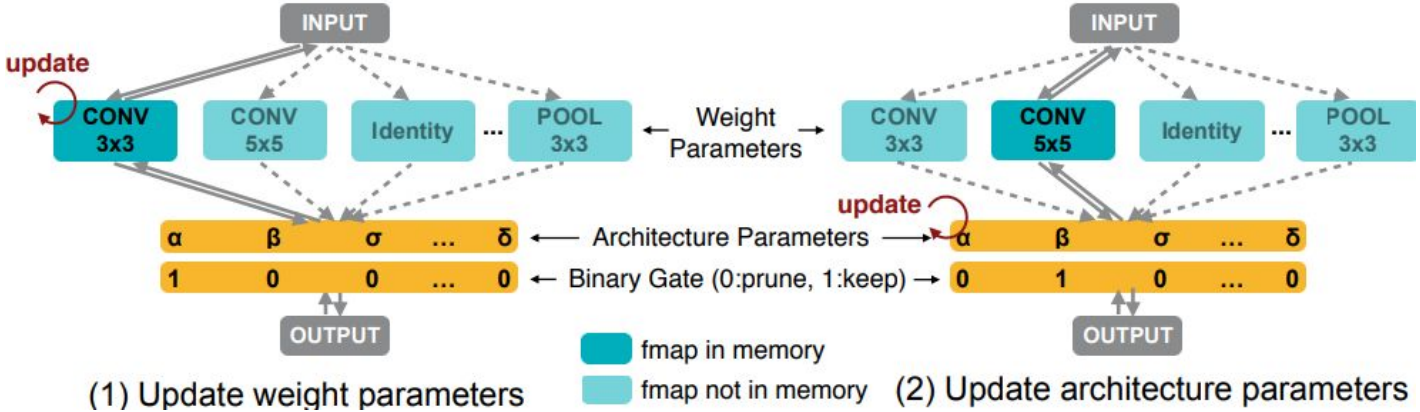
Liu, Hanxiao, Karen Simonyan, and Yiming Yang. "Darts: Differentiable architecture search." *arXiv preprint arXiv:1806.09055* (2018).

ProxylessNAS

$$m_{\mathcal{O}}^{\text{One-Shot}}(x) = \sum_{i=1}^{N} o_i(x), \qquad m_{\mathcal{O}}^{\text{DARTS}}(x) = \sum_{i=1}^{N} p_i o_i(x) = \sum_{i=1}^{N} \frac{\exp(\alpha_i)}{\sum_j \exp(\alpha_j)} o_i(x)$$

Cai, Han, Ligeng Zhu, and Song Han. "Proxylessnas: Direct neural architecture search on target task and hardware." *arXiv preprint arXiv:1812.00332* (2018).

ProxylessNAS



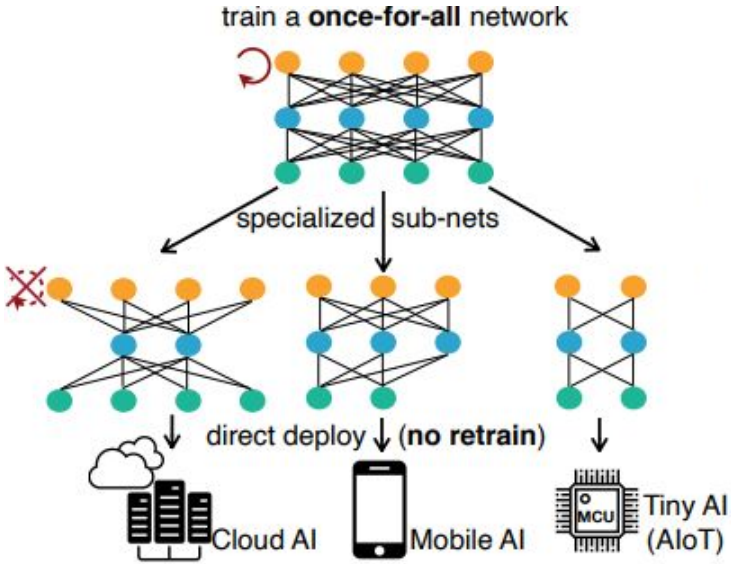
$$\mathbb{E}[\text{Latency}] = \alpha \times F(\text{conv\_3x3}) + \beta \times F(\text{conv\_5x5}) + \sigma \times F(\text{identity}) + \ldots\ldots + \zeta \times F(\text{pool\_3x3})$$

$$\mathbb{E}[\text{latency}] = \sum_i \mathbb{E}[\text{latency}_i]$$

$$Loss = Loss_{CE} + \lambda_1 ||w||_2^2 + \lambda_2 \mathbb{E}[\text{latency}]$$

NYU SAI LAB

Once-For-All: Train One Network And Specialize It for Efficient Deployment

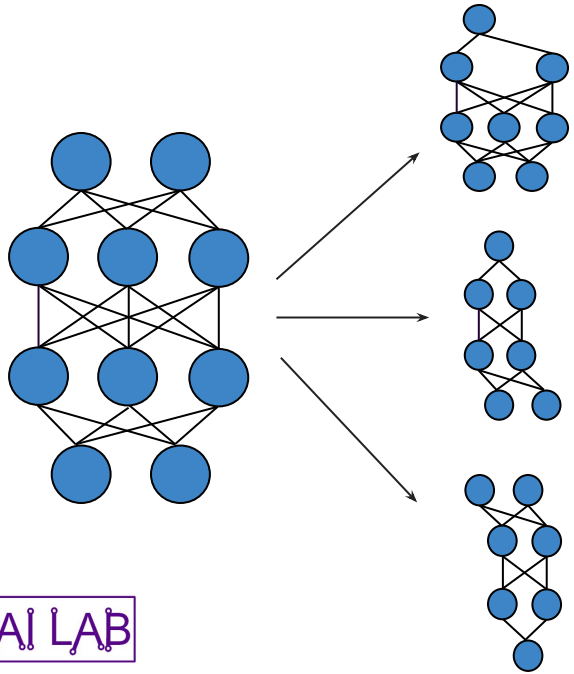


- A supernet is train which contained multiple smaller subnetworks.
- All the subnetworks are trained jointly.

NYU SAI LAB

Cai, Han, et al. "Once-for-all: Train one network and specialize it for efficient deployment." *arXiv preprint arXiv:1908.09791* (2019).

Once-For-All: Train One Network And Specialize It for Efficient Deployment



- During the forward pass, a subnetwork is sampled.
- All the weight within the subnetwork is trained in this iteration.
- All the subnetworks are nested within the OFA.
- A DNN is trained to search for the subnetwork.
  - Takes the neural network architecture as input.
  - Return the corresponding subnetwork that satisfies the criteria.

NYU SAI LAB

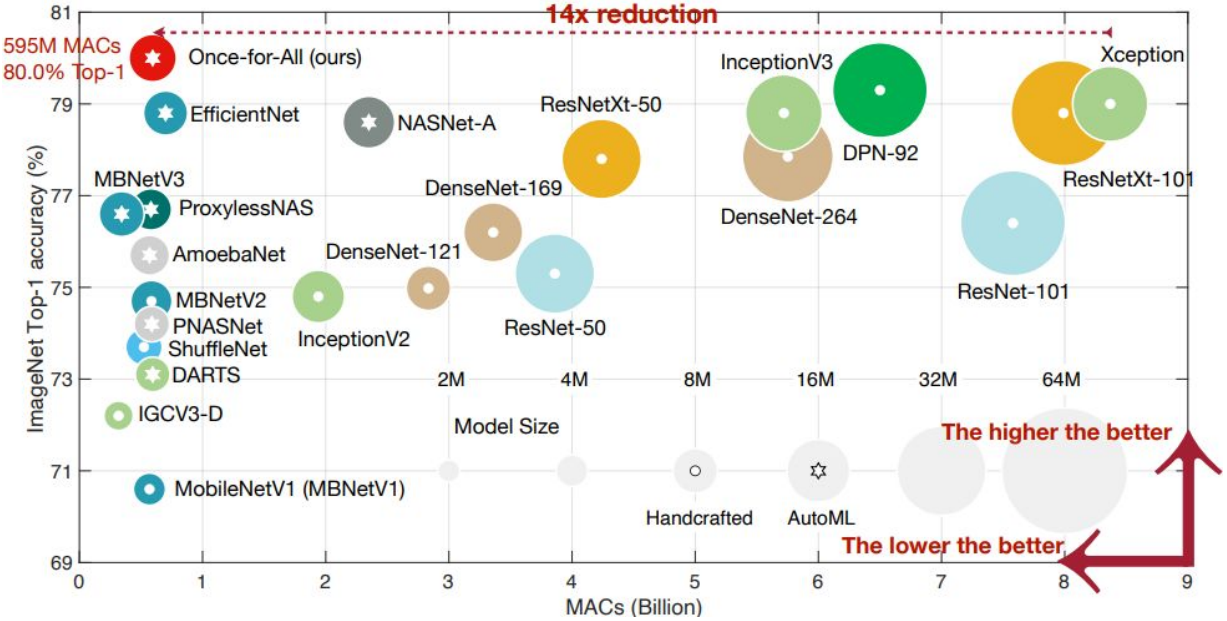
Cai, Han, et al. "Once-for-all: Train one network and specialize it for efficient deployment." *arXiv preprint arXiv:1908.09791* (2019).

Once-For-All: Train One Network And Specialize It for Efficient Deployment

- The once-for-all network comprises many sub-networks of different sizes where small sub-networks are nested in large sub-networks.
- After OFA is trained, we build neural-network-twins to predict the latency and accuracy given a neural network architecture.
- We randomly sample 16K sub-networks with different architectures and input image sizes, then measure their accuracy on 10K validation images sampled from the original training set.

Cai, Han, et al. "Once-for-all: Train one network and specialize it for efficient deployment." *arXiv preprint arXiv:1908.09791* (2019).

Once-For-All: Train One Network And Specialize It for Efficient Deployment



NYU SAI LAB

Cai, Han, et al. "Once-for-all: Train one network and specialize it for efficient deployment." *arXiv preprint arXiv:1908.09791* (2019).

Once-For-All: Train One Network And Specialize It for Efficient Deployment

- OFA enables multiple DNN to be trained at once, rather than search for a DNN with specific target each time, therefore greatly saving the training cost

Model	ImageNet Top1 (%)	MACs
MobileNetV2 [31]	72.0	300M
MobileNetV2 #1200	73.5	300M
NASNet-A [44]	74.0	564M
DARTS [25]	73.1	595M
MnasNet [33]	74.0	317M
FBNet-C [36]	74.9	375M
ProxylessNAS [4]	74.6	320M
SinglePathNAS [8]	74.7	328M
AutoSlim [38]	74.2	305M
MobileNetV3-Large [15]	75.2	219M
OFA w/o PS	72.4	235M
OFA w/ PS	**76.0**	230M
OFA w/ PS #25	**76.4**	230M
OFA w/ PS #75	**76.9**	230M
$\text{OFA}_{\text{Large}}$ w/ PS #75	**80.0**	595M

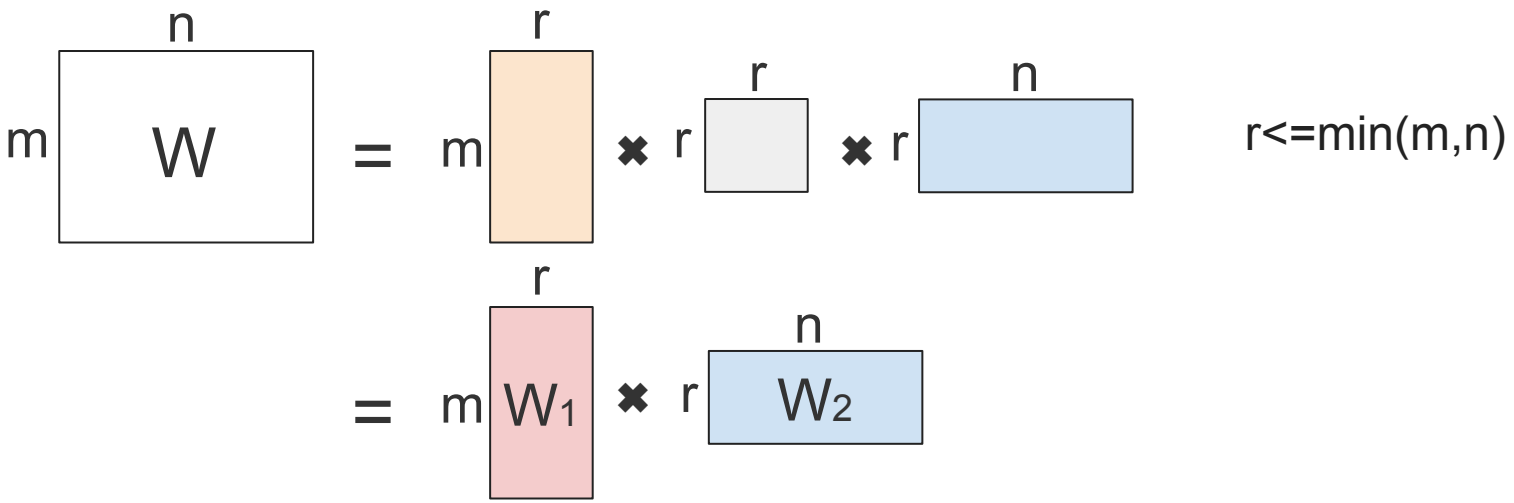
NYU SAI LAB

Topics

- Distillation
- Neural architecture search (NAS)
- Low-rank factorization
- Dynamic / Conditional Computing

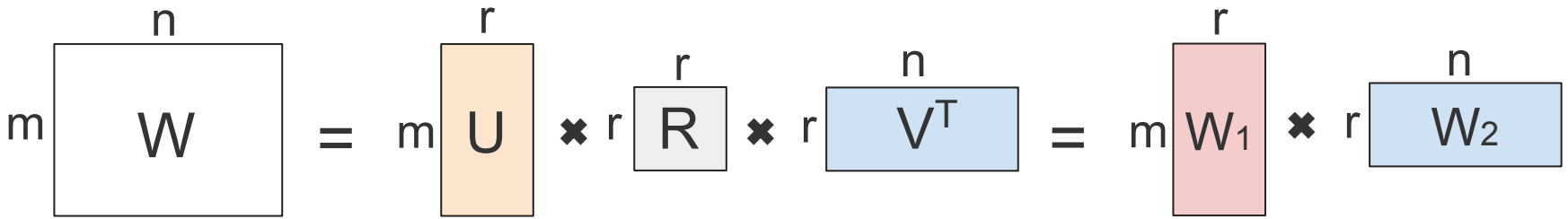
Low Rank Optimization for DNN Efficiency

- Weight tensors can be decomposed into:



- We can train the W_1 and W_2 in the DNN instead of W.
- Less storage is required.

Singular Value Decomposition



- W: Input matrix
  - m×n matrix
- V (n×r matrix) contains the orthonormal eigenvectors of W^TW
- U (m×r matrix) contains the orthonormal eigenvectors of WW^T
- R: Singular value matrix
  - r×r diagonal matrix, r is the rank of W

Before: mn

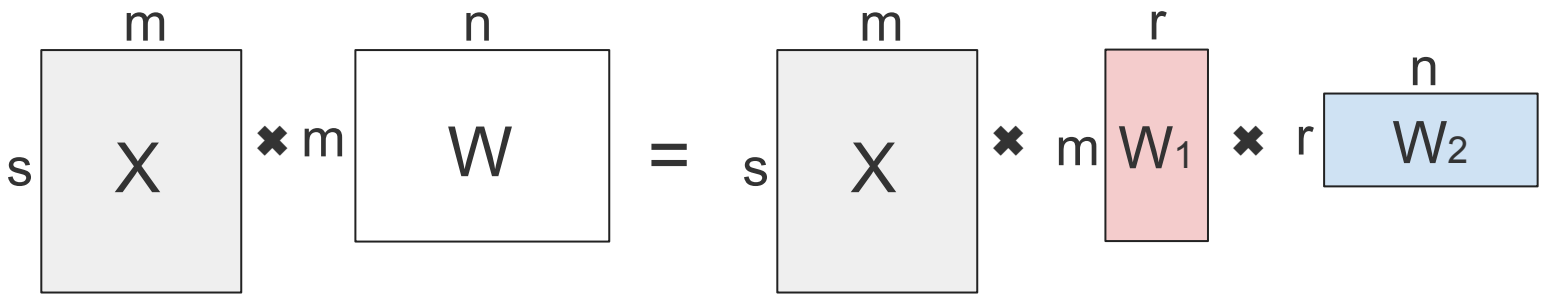
After: mr + rn = r(m+n) = min(m,n)(m+n)

assume W is full rank

- Without rank truncation, the number of parameters increases.

NYU SAI LAB

Singular Value Decomposition



Computational cost=smn

Computational cost = smr + snr
= s(m+n)r
= s(m+n)min(m,n)

- Without rank truncation, the computational cost will increase.

Singular Value Decomposition

$$A = \begin{bmatrix} 3 & 2 \\ 2 & 3 \\ 1 & 0 \end{bmatrix} \Rightarrow \boxed{A = U\Sigma V^T \quad \Sigma = \begin{bmatrix} 5.24 & 0 \\ 0 & 1.31 \\ 0 & 0 \end{bmatrix} \quad \sigma_1 = 5.24, \quad \sigma_2 = 1.31}$$

Truncate the smaller singular value

$$A_1 \approx \begin{bmatrix} 2.9 & 2.7 \\ 2.7 & 2.9 \\ 1.3 & 1.2 \end{bmatrix} \Leftarrow A_1 = \sigma_1 u_1 v_1^T$$

- We can truncate the smaller singular values in order to reduce the computation and storage cost.

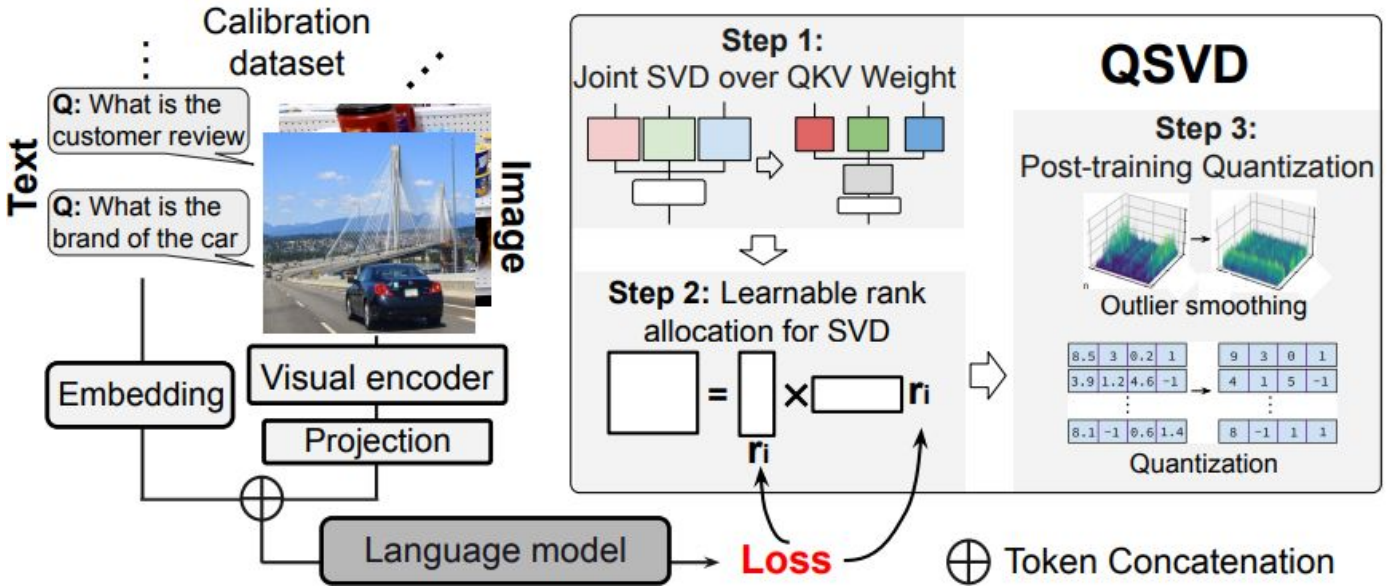
Singular Value Decomposition

$$\min_{W_1, W_2} \sum_{X \in \mathcal{D}} ||XW - XW_1W_2||^2$$

$$W_1 \in \mathcal{R}^{m \times r} \qquad W_2 \in \mathcal{R}^{r \times n}$$

- We can solve the above problems with local finetuning to generate a better answer.

QSVD: Efficient SVD and Quantization for VLM



- We propose an efficient joint SVD and quantization approach to facilitate the execution of VLM.

NYU SAI LAB

Haiyu Wang, et.al . "QSVD: Efficient Low-rank Approximation for Unified Query-Key-Value Weight Compression in Low-Precision Vision-Language Models." *NeurIPS* (2025).

Other Types of Decomposition Methods

- cp-decomposition
  - Lebedev, Vadim, et al. "Speeding-up convolutional neural networks using fine-tuned cp-decomposition." *arXiv preprint arXiv:1412.6553* (2014).
- Tucker decomposition
  - Kim, Yong-Deok, et al. "Compression of deep convolutional neural networks for fast and low power mobile applications." *arXiv preprint arXiv:1511.06530* (2015).
- Each method has its own tradeoff between computational efficiency and Accuracy.

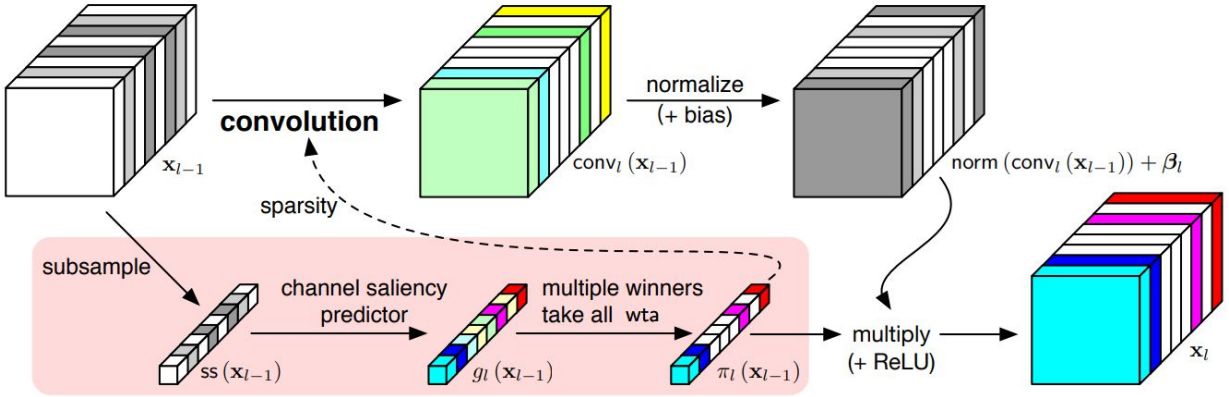
Lebedev, Vadim, et al. "Speeding-up convolutional neural networks using fine-tuned cp-decomposition." *arXiv preprint arXiv:1412.6553* (2014).

Topics

- Distillation
- Neural architecture search (NAS)
- Low-rank factorization
- Dynamic / Conditional Computation

Dynamic / Conditional Computation

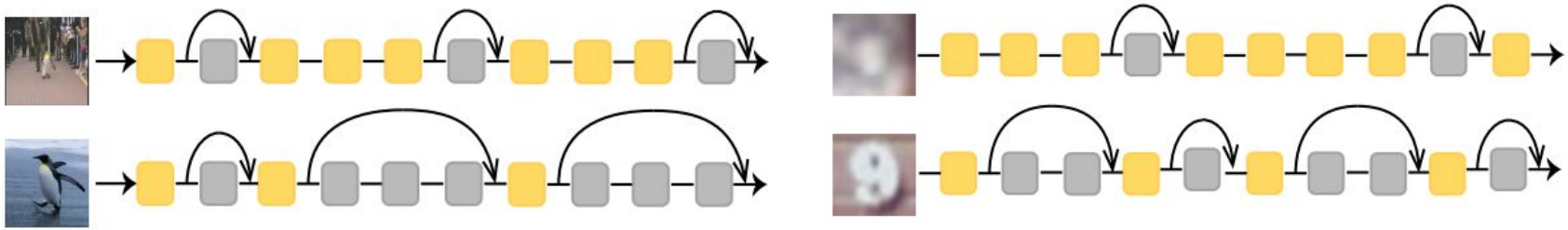
- Activates only parts of the network depending on input.



- a channel-wise importance measure is generated.

Gao, Xitong, et al. "Dynamic channel pruning: Feature boosting and suppression." *arXiv preprint arXiv:1810.05331* (2018).

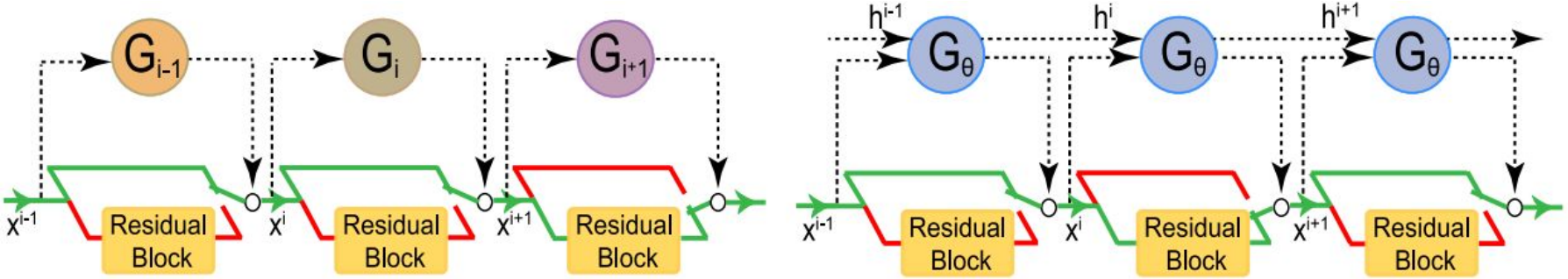
SkipNet



- We introduce SkipNet, a modified residual network, that uses a gating network to selectively skip convolutional blocks based on the activations of the previous layer.

Wang, Xin, et al. "Skipnet: Learning dynamic routing in convolutional networks." *Proceedings of the European conference on computer vision (ECCV)*. 2018.

Dynamic Gating



$$G_{\text{relax}}(\mathbf{x}) = \begin{cases} \mathbb{I}(S(\mathbf{x}) \geq 0.5), & \text{forward pass} \\ S(\mathbf{x}), & \text{backward pass} \end{cases}$$

- The forward pass produce the binary results. The backward pass applies STE to skip the binary function

BranchyNet

- Data samples are not equal in their recognition difficulties.
- For the easy samples, they only needs to be processed with a few layers before generating the correct results.

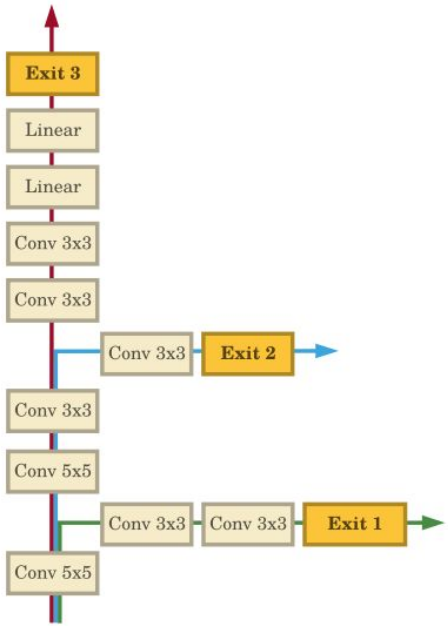
NYU SAI LAB

Teerapittayanon, Surat, Bradley McDanel, and Hsiang-Tsung Kung. "Branchynet: Fast inference via early exiting from deep neural networks." *2016 23rd international conference on pattern recognition (ICPR)*. IEEE, 2016.

BranchyNet

- During Inference, a confidence score is computed at each exit point, if greater than a predefined threshold, then the output is computed locally, leading to a faster inference.

- The confidence score is defined as: $\text{entropy}(\boldsymbol{y}) = \sum_{c \in \mathcal{C}} y_c \log y_c,$



NYU SAI LAB

Teerapittayanon, Surat, Bradley McDanel, and Hsiang-Tsung Kung. "Branchynet: Fast inference via early exiting from deep neural networks." *2016 23rd international conference on pattern recognition (ICPR)*. IEEE, 2016.

Presentations

- Learned Step Size Quantization (Weikai Qu, Peiheng Yao, Hanyu Da)
- Quantization & Training of Neural Nets for Efficient Int-Arithmetic-Only Inference (Hao Yang, Shiyu Zhang)