On Merging MobileNets for Efficient Multitask Inference

Cheng-En Wu, **Yi-Ming Chan**, and Chu-Song Chen

Institute of Information Science, Academia Sinica, Taiwan

MOST Joint Research Center for AI Technology
and All Vista Healthcare

Outline

- Introduction
- Related Work
- Merging MobileNets
- End-to-end Fine-Tuning
- Experiments
- Conclusion

Introduction

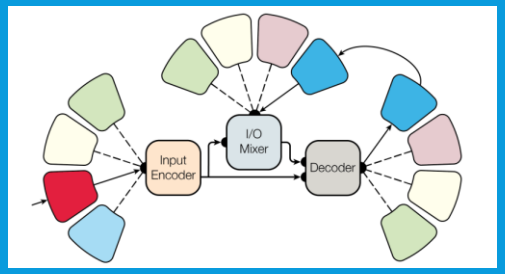
- Deep neural networks got success in computer vision, medical imaging, and multimedia processing.
- We usually train different networks for different tasks to make them behave well for each specific purpose.
- In practical applications, however, it is common to handle multiple tasks simultaneously, resulting in a high demand for resources.
- It becomes a crucial problem to effectively integrate multiple neural networks in the training and inferencing stage.

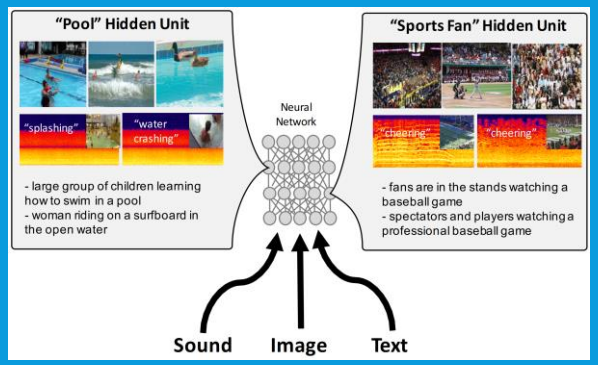
Introduction

- To reduce the computational cost, compact network architectures are developed
  - MobileNet [Howard *et al*., 2017]
  - ShuffleNet [Zhang *et al.*, 2018]
  - XNOR-Net [Rastegari *et al.*, 2016]
- Although ShuffleNet or XNOR-Net are compact and efficient, their accuracy drop a lot.
- MobileNet is one of the best model with balanced speed and accuracy, and thus is chosen as our backbone networks.

Related Works

- Multi-task Deep Models
  - In [1], Multi-Model architecture is introduced.
    - Convert different inputs by encoder
    - Consider complex short cut connection
    - Decode multiple tasks with a decoder
  - In [2], representation is aligned to share across modalities.
- Nevertheless, the "the-learn-them-all" approaches pay cumbersome training effort and intensive inference complexity.



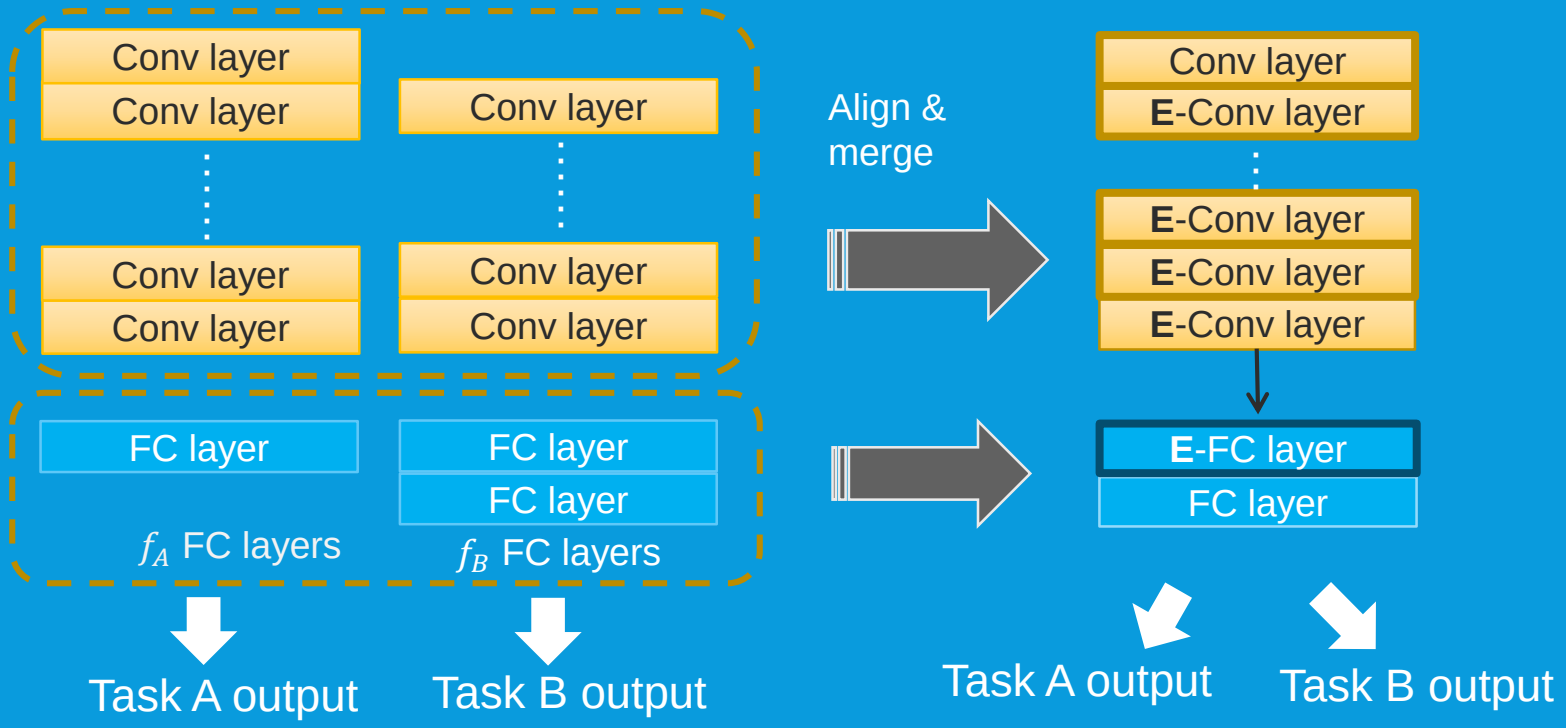


[1] L. Kaiser *et al.*, "One Model To Learn Them All," *CoRR,* vol. abs/1706.05137, 2017.

[2] Y. Aytar, C. Vondrick, and A. Torralba, "See, hear, and read: Deep aligned representations," *arXiv preprint arXiv:1706.00932,* 2017.

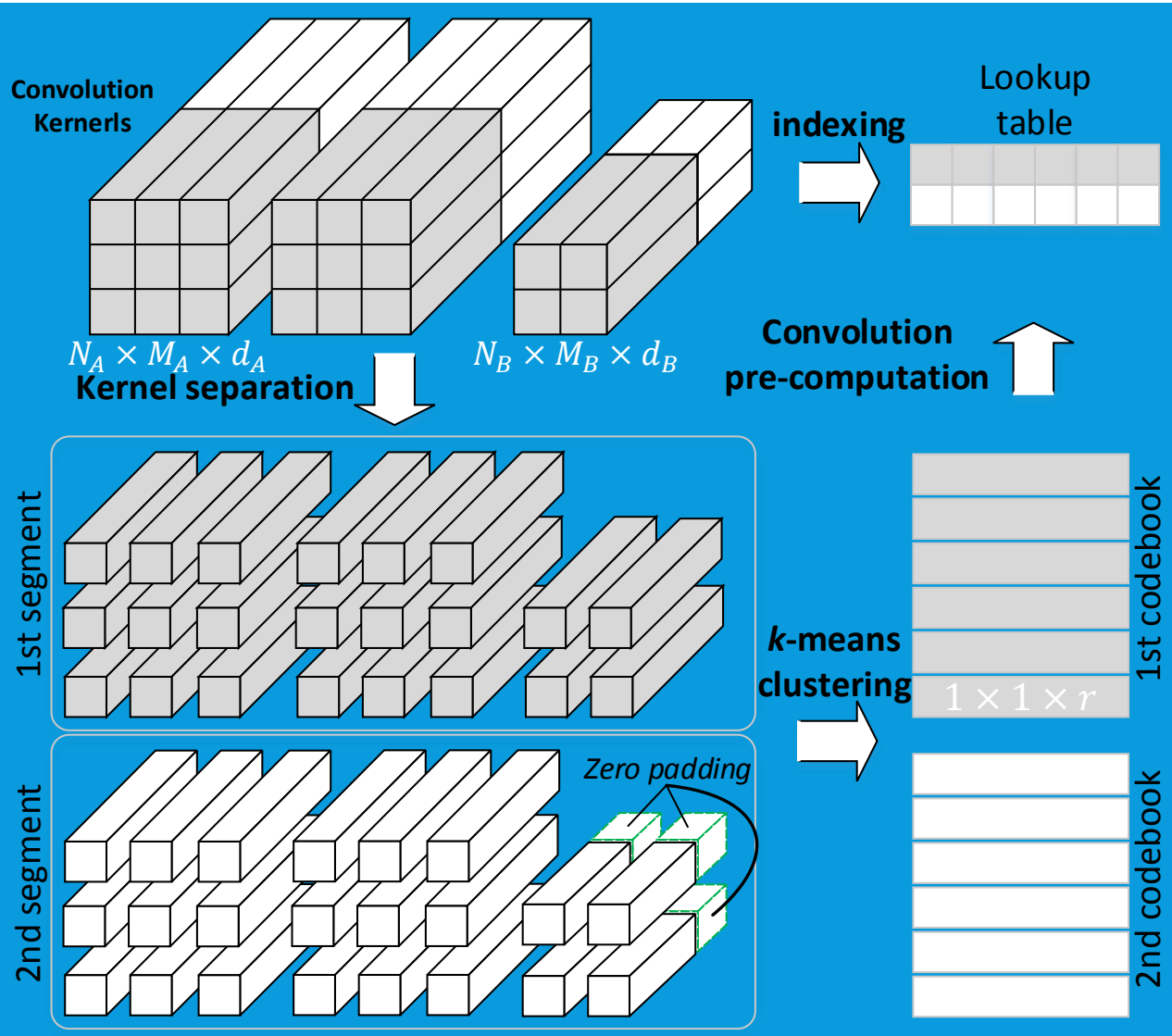
Related Works

- In our previous work [1], our system merged well-trained models using vector quantization technique.



[1] Y.-M. Chou, Y.-M. Chan, J.-H. Lee, C.-Y. Chiu, and C.-S. Chen, "Unifying and merging well-trained deep neural networks for inference stage," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018, pp. 2049-2056

Related Works

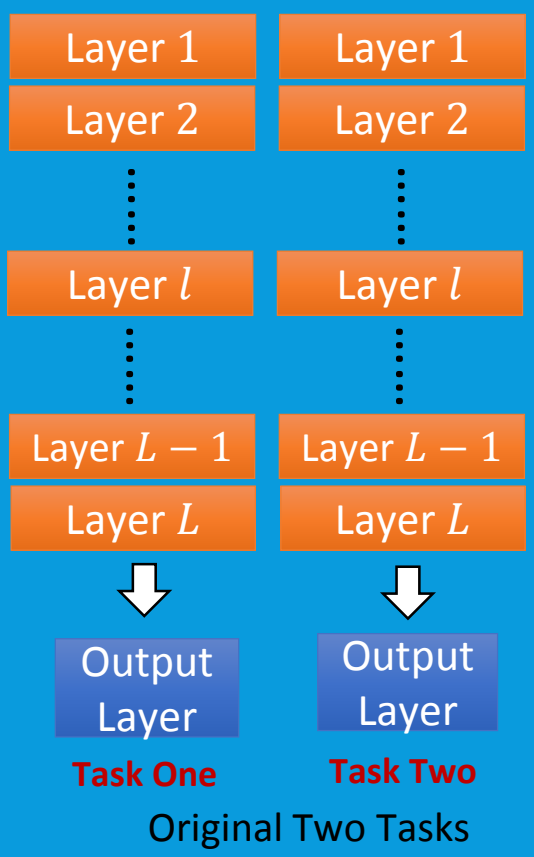


Related Works

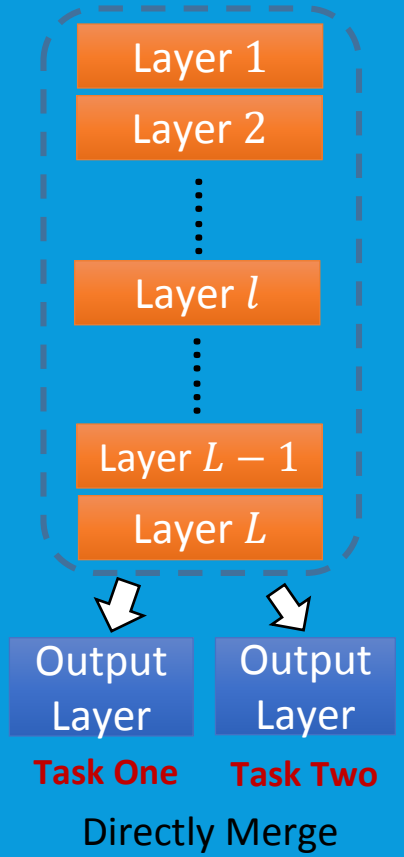
- Although our previous work can simultaneously achieve model speedup and compression with negligible accuracy drop, the modified layers are not supported by deep learning frameworks like TensorFlow or pyTorch, etc.
  - The modified layers require 1×1 convolutions and extra table lookups with value summations.
  - Currently, it is achieved with “hand-made” C++ code under CPU mode only.
  - Only basic layer operations (for AlexNet, and VGG16) are supported right now.
- On the contrary, this work can take the advantages of TensorFlow to merge two networks (MobileNet).

Merging MobileNets

- Naïve solution (baseline)
  - Directly train a shared network with two different output layers.



Original Two Tasks

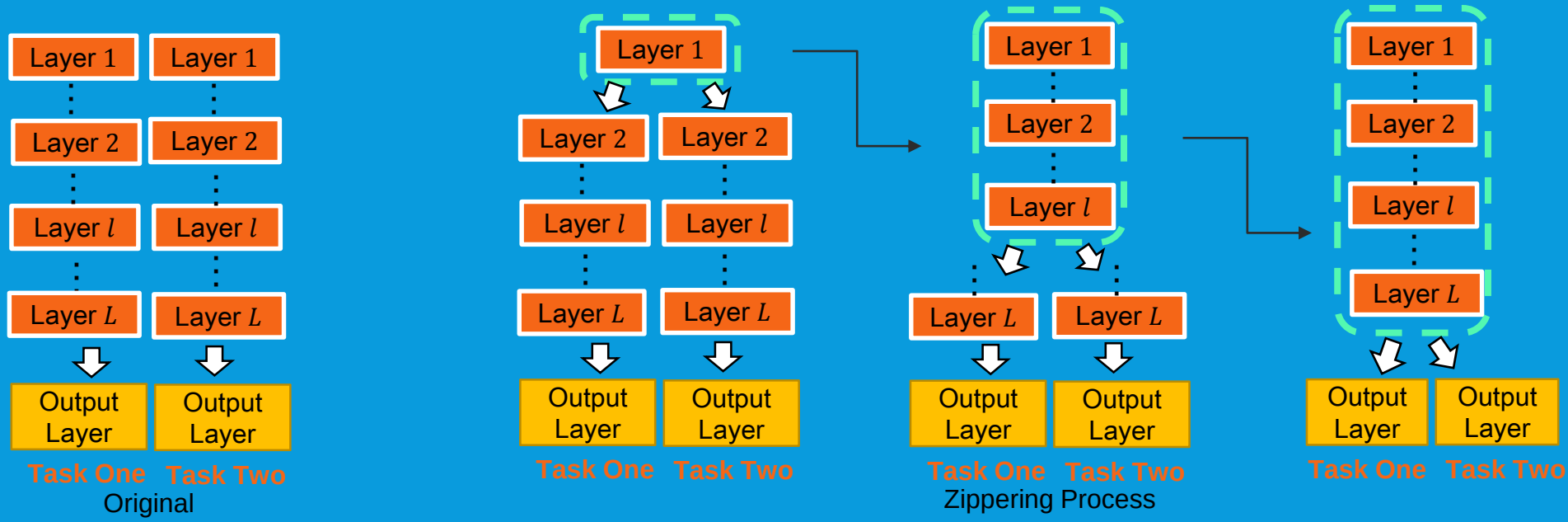


Directly Merge

Easy to implement but initialization of weight may be biased.

Merging MobileNets

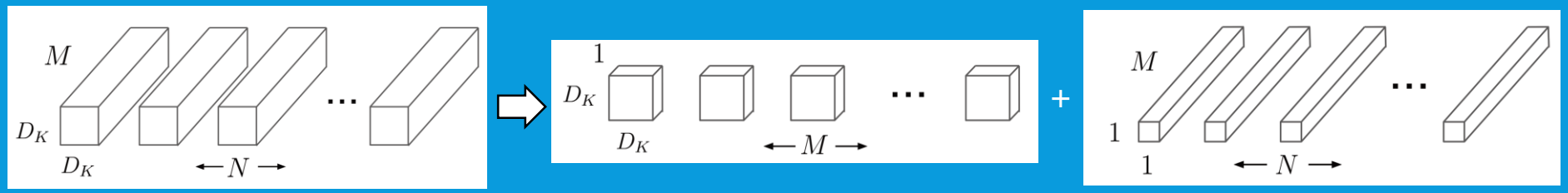
- "Zippering" Process
  - Iteratively merge two networks from the input to output.
  - Merge and initialize the layer.
  - Calibrate merged weight to restore the performance.



Merging MobileNets

- Implementation Details
  - Only point-wise convolution layers in MobieNet architecture are merged, because
    - The computational cost of point-wise convolution is much greater than that of depth-wise convolution layer.
    - The depth-wise convolution serves as main spatial feature extractor.

Depth-wise separable convolution in MobileNet



Original Convolution Filters | Depth-wise Convolution Filters | Point-wise Convolution Filters

Weight Initialization and Calibration

- Weight initialization is important for training performance
- For merging two MobileNets A and B, potential solutions are:
  - Initialized by $\mathbf{W}_A$
  - Initialized by $\mathbf{W}_B$
  - Random
  - Initialized by arithmetic mean of each filter of the layer

$$\boldsymbol{\mu}_i = \frac{\boldsymbol{W}_{\boldsymbol{A_i}} + \boldsymbol{W}_{B_i}}{2}, i = \{1, \dots, C\}$$

where C is number of output Channel

Simple, but effective!

Weight Calibration Training

- Original models serve as teacher networks
  - When applying the input x_I to the model A (or B), the output of every layer in the merged model should be close to the output of the associated layer in A (or B)
- Two types of minimization terms in calibration training
  - **Classification (or regression) error** in the original tasks A and B.
  - **Layer-wise output mismatch error**
- L_1 loss is used
- Student (merged network) can learn well even with few iterations.
- Implemented using Tensorflow framework.

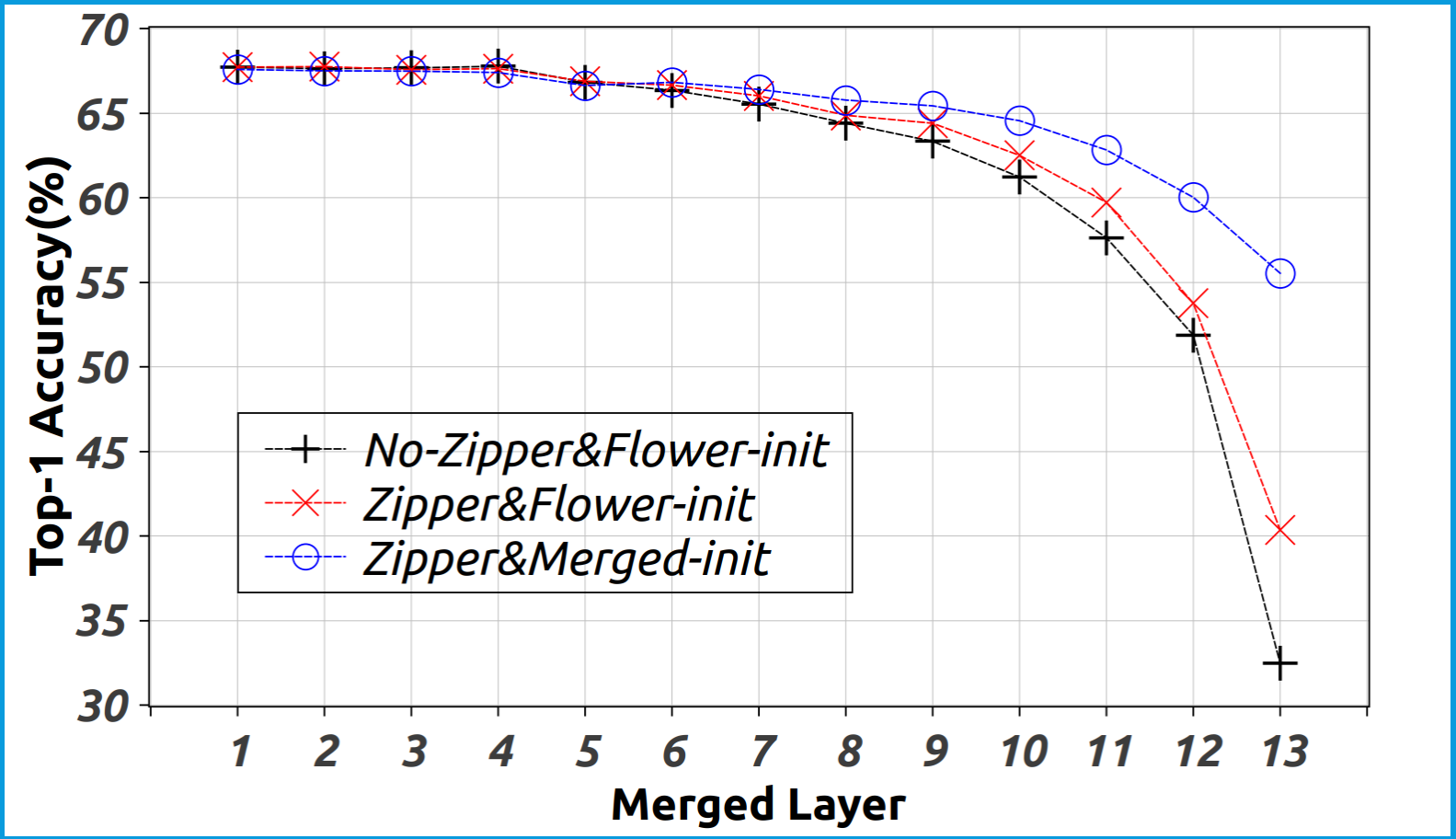
Experiments

- Datasets
  - ImageNet: General image classification
  - DeepFashion: Clothing classification
  - CUBS Birds: Birds classification
  - Flowers: Flowers classification

Name	Classes	Training Set	Testing Set
ImageNet	1000	1,281,144	50,000
DeepFashion	50	289,222	40,000
CUBS Birds	196	5,994	5,794
Flowers	102	2,040	6,149

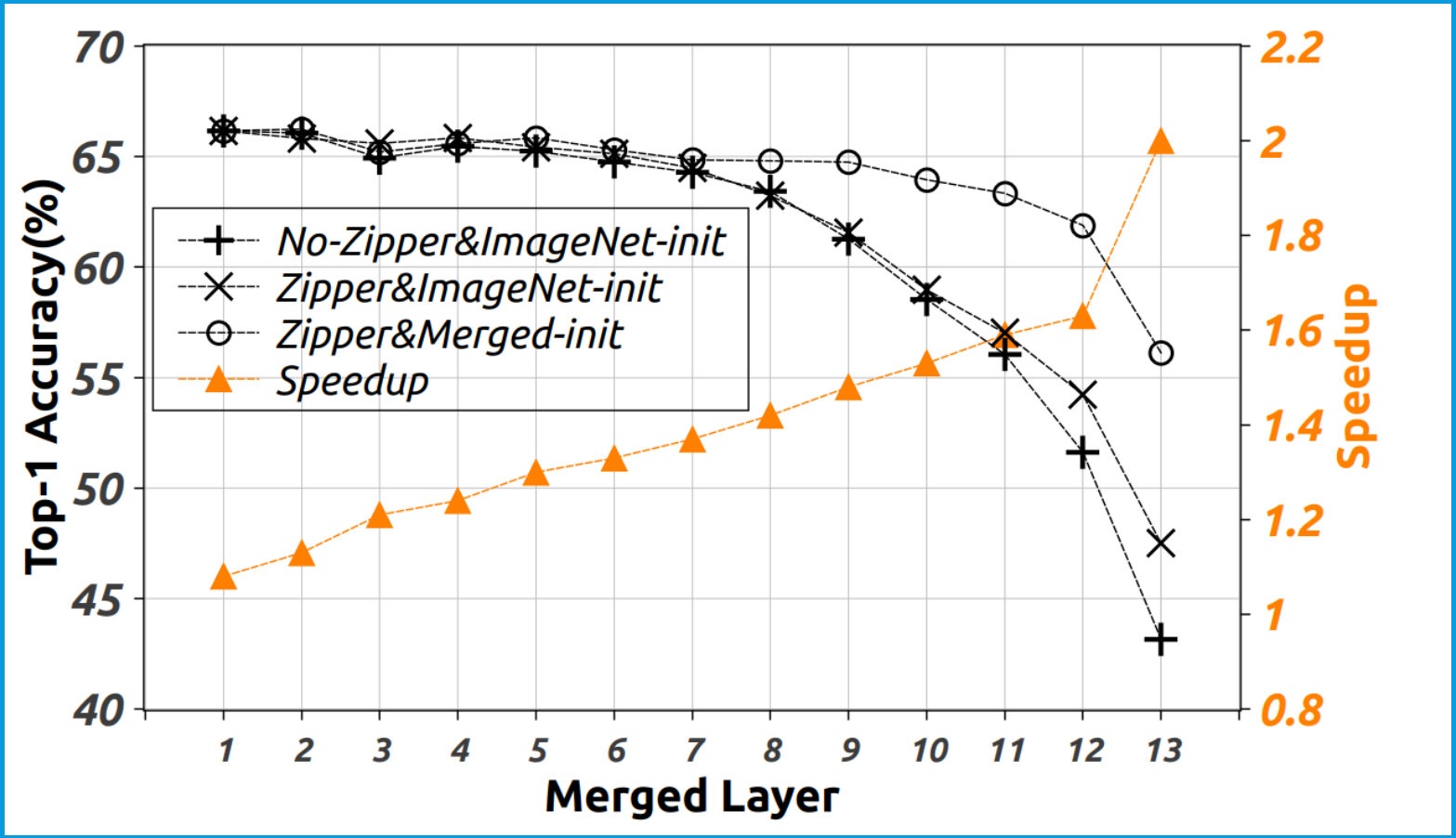
Experiments

- Merge of Flower and CUBS MobileNets
  - Top-1 Classification Accuracy in CUBS Birds Dataset



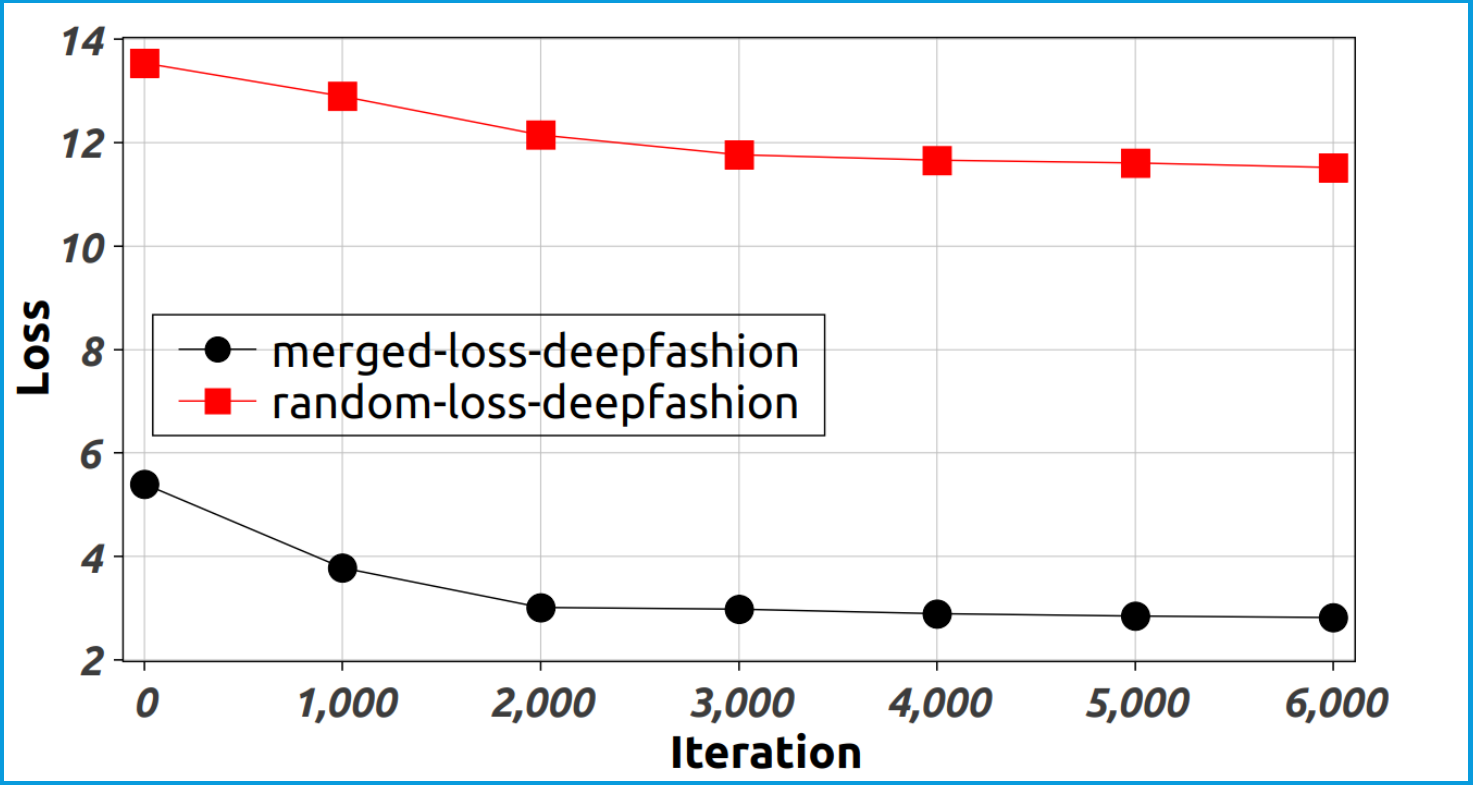
Experiments

- Merge of ImageNet and DeepFashion
  - Accuracy and speedup on DeepFashion dataset



Experiments

- Convergent speed of different initialization method
  - Merged of DeepFashion and ImageNet
  - Loss on DeepFashion dataset



Experiments

- Details of speedup, compression rate, and accuracy of merging ImageNet and DeepFashion or CUBS and Flowers.

Merged Layer	Comp.	Speedup	Runtime memory usage Comp.	Acc.(%) (ImageNet[1])	Acc.(%) (DeepFashion[1])	Acc.(%) (CUBS[2])	Acc.(%) (Flowers[2])
0	1.000	1.00	1.00	71.02	66.21	68.01	91.70
1(First)	1.000	1.08	1.09	70.55	66.24	67.58	91.63
1~2	1.002	1.13	1.14	70.22	66.17	67.50	91.33
1~3	1.004	1.21	1.19	69.65	65.95	67.48	90.82
1~4	1.009	1.24	1.23	69.46	65.59	67.40	91.02
1~5	1.020	1.30	1.27	68.94	65.85	66.63	91.18
1~6	1.042	1.33	1.28	68.53	65.31	66.82	90.75
1~7	1.089	1.37	1.30	68.25	64.85	66.40	90.53
1~8	1.140	1.42	1.32	67.75	64.81	65.77	90.16
1~9	1.196	1.48	1.34	67.34	64.75	65.43	89.74
1~10	1.258	1.53	1.37	66.66	63.95	64.55	88.40
1~11	1.258	1.53	1.39	65.70	63.34	62.82	86.66
1~12	1.258	1.53	1.40	64.35	61.89	60.02	84.00
1~13(All)	1.258	1.53	1.41	61.63	56.12	55.53	80.99

Conclusion

- We present a method that can merge CNNs into a single but more compact one.
- The “zippering-process” of merging two architecture identical MobileNet is proposed.
- The simple-but-effective weight initialization can shorten fine-tune time to restore the performance.
- Experimental results show that the merged model can be take the advantage of public deep learning framework with satisfactory speedup and model compression.
- Future work will be the merging of different network architecture.