Divide and Conquer: Leveraging Intermediate Feature Representations for Quantized Training of Neural Networks

Ahmed T. Elthakeb [1] **Prannoy Pilligundla** [1] **Hadi Esmaeilzadeh** [1]

Abstract

The deep layers of modern neural networks extract a rather rich set of features as an input propagates through the network. This paper sets out to harvest these rich intermediate representations for quantization with minimal accuracy loss while significantly reducing the memory footprint and compute intensity of the DNN. This paper utilizes knowledge distillation through teacher-student paradigm (Hinton et al., 2015) in a novel setting that exploits the feature extraction capability of DNNs for higher-accuracy quantization. As such, our algorithm logically divides a pretrained full-precision DNN to multiple sections, each of which exposes intermediate features to train a team of students independently in the quantized domain. This divide and conquer strategy, in fact, makes the training of each student section possible in isolation while all these independently trained sections are later stitched together to form the equivalent fully quantized network. Experiments on various DNNs (LeNet, ResNet-20, SVHN and VGG-11) show that, on average, this approach—called **DCQ** (**D**ivide and **C**onquer **Q**uantization)—achieves on average 9.7% accuracy improvement to a state-of-the-art quantized training technique, DoReFa (Zhou et al., 2016) for binary and ternary networks.

1. Introduction

Today deep learning, with its superior performance, dominates a wide range of real life inference tasks including image recognition, voice assistants, and natural language processing (Hauswald et al., 2015; Krizhevsky et al., 2012; LeCun et al., 2015; 1989). However, the shear complexity of deep learning models and the associated heavy compute and memory requirement appears as a major challenge as the demand for such services rapidly scale. Quantization, which can reduce the complexity of each operation as well as the overall storage requirements of the DNN, has proven to be a promising path forward. Nevertheless, quantization requires carefully tailored training and recovery algorithms (Courbariaux et al., 2015; Gupta et al., 2015; Hubara et al., 2017a; Zhou et al., 2017; 2016) to even partially overcome its losses in accuracy.

In this paper, we set out to devise an algorithm that enables quantization with much less accuracy degradation. The key insight is that the intermediate layers of a deep network already extract a very rich set of features and these intermediate representations can be used to train/teach a quantized network more effectively. To that end, we define a new sectioning based approach towards knowledge distillation through teacher-student paradigm (Hinton et al., 2015; Bucila et al., 2006) focusing on teaching the knowledge of intermediate features to a corresponding quantized student.

We evaluate DCQ using a variety of DNNs including LeNet, ResNet-20, SVHN and VGG-11 with binary and ternary weights. Across all the quantization levels, DCQ, on average, delivers 7.7% higher accuracy than DoReFa (Zhou et al., 2016), the state-of-the-art quantization algorithm. These encouraging results suggest that leveraging the inherent feature extraction ability of DNNs for knowledge distillation can lead to significant improvement in their efficiency, reducing their bitwidth in this particular case.

2. Related Work

Knowledge distillation. Knowledge distillation (Hinton et al., 2015) is proposed to attain a smaller/shallower neural network (student) from one or an ensemble of bigger deep networks (teacher). The student network is trained on a softened version of the *final* output of teacher(s) (Bucila et al., 2006). FITNETS (Romero et al., 2015) extends knowledge distillation by extracting a hint from the teacher

[1]Alternative Computing Technologies (ACT) Lab, University of California San Diego, USA. Correspondence to: Ahmed T. Elthakeb <a1yousse@eng.ucsd.edu>.

ICML 2019 Workshop on Understanding and Improving Generalization in Deep Learning, Long Beach, California, 2019.

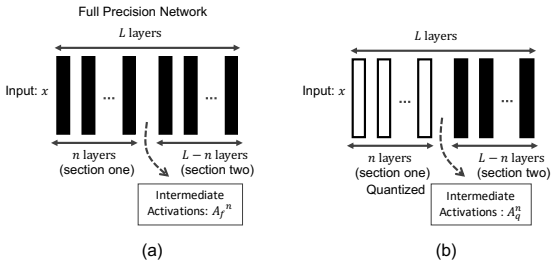

Figure 1. DCQ two stage split example

to train even a deeper but thinner student. The hint is intermediate feature representation of the teacher, that is used as a regularizer to pretrain the first few layers of the deep and thin student network. After the pretraining phase, the full knowledge distillation is used to finish the training of the student. FITNETS does not explore hints from more than one intermediate layer of the teacher. Furthermore, FITNETS applies the knowledge distillation pass over the entire student network at once. FITNETS are a complementary approach to our sectional knowledge distillation and similar hints can be utilized for each section. Nonetheless, the following discusses the differences. In contrast to this technique, DCQ (1) partitions the neural network to multiple independent sections and (2) applies knowledge distillation to each section in isolation and trains them independently, (3) After the sections are trained through knowledge distillation, they are put together instead of applying another phase of training as done in FITNETS. (4) Finally, the objective differ as the knowledge distillation and FITNETS aim to compress the network while DCQ quantizes it. Knowledge distillation is also used for training a lower bitwidth student network from a full-precision teacher (Mishra & Marr, 2018). However, this work does not partition the network as DCQ does and also does not utilize teacher's intermediate layers.

Other quantization techniques. Multiple techniques have been proposed for low bidwidth/quantized training of neural networks. DoReFa-Net (Zhou et al., 2016), WRPN (Mishra et al., 2018), TTQ (Zhu et al., 2017) and PACT (Choi et al., 2018) propose different techniques to quantize either weights, activations or both. There have also been lot of efforts (Rastegari et al., 2016; Li & Liu, 2016; Hubara et al., 2017b) to binarize neural networks at the cost of some accuracy loss. However, these inspiring efforts do not introduce sectioning nor they leverage knowledge distillation in the context of either quantization or binarizing the neural networks.

3. DCQ: Divide and Conquer for Quantization

Overview. We take inspiration from knowledge distillation and apply it to the context of quantization by proposing a novel technique dubbed DCQ. The main intuition behind DCQ is that a deeply quantized network can achieve accuracies similar to full precision networks if intermediate layers

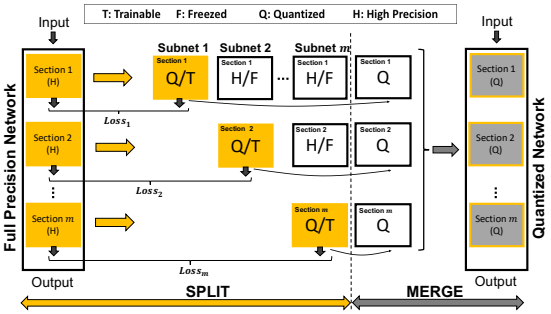

Figure 2. (a) Divide and Conquer approach overview showing SPLIT phase; dividing the teacher full precision network into smaller subnetworks, and MERGE; by combining the training results of each subnetwork to form a fully quantized network, (b) Detailed training procedure

of the quantized network can retain the feature representation that was learnt by the full precision network. To this end, DCQ splits the quantized network and full precision network into multiple small sections and trains each section individually by means of partial backpropagation so that every section of the quantized network learns and represents similar features as the corresponding section in the full precision network. This section describes different steps and rationale of our technique in more detail.

3.1. Matching activations for intermediate layers

Figure 1 (a) shows a full precision network where as Figure 1 (b) is a deeply quantized version of the same network where first n layers are quantized and the remaining $L-n$ layers are at full precision. When we pass the same input image x to both these networks, if the output activations of layer n for full precision network i.e A_f^n are equivalent to the output activations of layer n for the semi-quantized network, A_q^n, then both the networks classify the input to a same class because rest of the $L-n$ layers are same for both the networks and their input activations are same as well. Therefore, if both these networks shown in Figure 1 (a) and (b), have similar output activations for all the input images, then the network with first n layers quantized has learnt to represent the same features as the first n layers of the original network and it will have the same classification accuracy as the full precision network. This is the underlying principle for our proposed quantization technique DCQ. In the above example, the network was split into two sections of n and $L-n$ layers, instead DCQ splits the original network into multiple sections and trains those sections individually to output same activations as the corresponding section in the full precision network.

3.2. Splitting, training and merging

Splitting the full precision network. As described in Section 3.1, DCQ splits the original network into multiple sections and trains them in isolation and in parallel. Figure 2

shows an overview of the entire process. As shown in the figure, after splitting the full precision network into m sub sections, DCQ quantizes and trains these subsections independently. After training, DCQ puts them all together again to get the deeply quantized version of the entire original network. As discussed in Section 3.1, because each of these sections is trained to capture the same features as the full precision network, although these sections are trained independently, they can be put together in the end to give similar accuracy as the full precision network.

Training the sub networks. As Figure 2 illustrates, we create m sub networks in order to train each of the m sections. For each sub section i, the sub network consists of all the sections before it. Subnet 1 column in Figure 2 shows a sub network for section 1. To train this section, the output activations of the quantized version of section 1 are compared with the output activations of the full precision version of section 1 and the loss is calculated accordingly. Section 3.3 gives more details on how the loss is calculated for each sub network. Similarly, Subnet 2 column box shows the sub network for section 2 and it comprises of both section 1 and section 2. Output activations of section 2 are used to calculate the loss in this case. Since section 2 is being trained in this sub network, weights for section 1 are frozen(not trainable) in this sub network and backpropagation based on the loss only affects section 2. Similarly there are sub networks for sections 3...m and the last sub network m is basically similar to the full precision network except that the section m is quantized and all the other sections from 1 to $m-1$ are frozen.

Merging the sections. After training all the sections, since each of these sections has been trained independently to learn the same features as the corresponding section of the fully precision network but with quantized weights, they can be put together to form a fully trained quantized network.

3.3. Loss function for training sub networks

Since DCQ aims to capture the intermediate features learnt by the full precision network, loss needs to be calculated based on the output activations of intermediate layers unlike the traditional loss which is calculated using the output of the final classification layer and the targets. In DCQ, we use the Poisson loss function as it is a measure of how the predicted distribution diverges from the expected distribution. Poisson loss is expressed by the following formula where $\hat{y}$ are the predicted activations i.e by output activations of the subnetwork, y are the output activations of that section of the full precision network and n is the total number of output activations for that sub network.

$$\mathscr{L} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}^{(i)} - y^{(i)}.\log(\hat{y}^{(i)}))$$

We provide results about the impact of using different loss formulations in Section 4.

Table 1. Summary of results comparing our approach (DCQ) to a conventional approach (DoReFa-Net) for different networks considering binary and ternary weight quantization

	Weight Quantization	LeNet On MNIST	ResNet-20 On CIFAR10	SVHN	VGG-11 On CIFAR10
		Acc.(%)	Acc.(%)	Acc.(%)	Acc.(%)
DoReFa	Binary {-1,1}	96.715	73.38	88.55	72.78
DoReFa + DCQ (Ours)		**99.283**	**90.52**	**93.21**	**87.48**
DoReFa	Ternary {-1,0,1}	98.917	83.52	91.72	81.98
DoReFa + DCQ (Ours)		**99.767**	**90.78**	**93.94**	**93.96**
	Full Precision	99.867	91.36	96.478	94.13

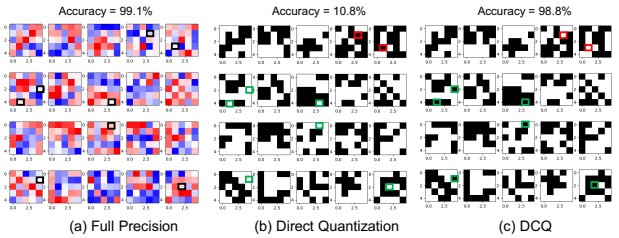

Figure 3. Visualization of a subset of weight kernels of the second convolutional layer of LeNet highlighting the differences between different versions of binary weight kernels: (a) Full precision weight kernels, (b) binary weight kernels upon direct binarization from full precision, (c) binary weight kernels obtained using our method DCQ

4. Experimental Results

Experimental Setup In this section, we evaluate the efficacy of our proposed approach on different datasets: MNIST, CIFAR10, and SVHN. We compare our approach to conventional end-to-end training approach and consider DoReFa-Net as our baseline. For all the experiments, we use an open source framework for compression, Distiller (Zmora et al., 2018). The reported accuracies for DoReFa-Net are with the built-in implementations in Distiller. While reporting accuracies in their paper, DoReFa-Net doesn't quantize first and last layers of the network whereas in our case, we quantize all the layers including the first and last layers. Because of this difference in quantization and using built-in implementation of Distiller, the accuracies we report might not exactly match the accuracies reported in their paper.

Performance for Binarization and Ternarization We focus on ternarization and binarization where weight tensors are quantized into {-1, 0, 1}, and {-1,1} respectively. Throughout the reported experiments, there is no per-layer nor per-channel scaling coefficients. Mere binary and ternary kernels are directly used for computations. Table 1 shows a summary of results comparing our approach, DCQ, to a conventional approach (DoReFa-Net) considering binary and ternary weight quantization for different networks: LeNet, ResNet-20, SVHN, and VGG-11. As seen in the table, under same training setup and parameters settings, DCQ significantly outperforms the conventional approach

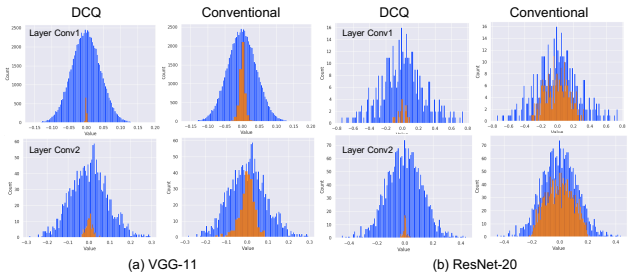

Figure 4. Weights histogram of first and second convolutional layers highlighting the altered portion of the trained binary weights relative to the directly binarized weighs using DCQ and the conventional end-to-end training method (a) VGG11, and (b) ResNet-20

and achieves a consistent improvements across the different networks. On average, DCQ achieves 9.8% accuracy improvement for binary weight networks, and 5.7% accuracy improvement for ternary weight networks.

Delving into the results, we posit the following elements behind the reported improvements. First, deep multi-hidden-layer neural networks are much more difficult to tackle as compared to shallower ones. Furthermore, end-to-end back-propagation can be inefficient (Jaderberg et al., 2017). Thus, adopting such divide and conquer approach yields simpler subproblems that are easier to optimize. Second, matching intermediate learning objectives also guides the optimization as compared to following a single global objective that indirectly specifies learning objectives to the intermediate layers.

DCQ vs Conventional Binary Kernels This subsection provides an analysis of our obtained binary weight kernels and sheds light on some interesting observations. We start by posing the following questions. (1) How are trained binary weight kernels different from just direct binarization from the original full precision weight kernels? and (2) Whether different training algorithms can yield qualitatively different binary weight kernels?

Figure 3 shows a visualization of a subset of weight kernels from the second conv. layer of LeNet. (a) is the original full precision kernels, (b) direct binarization of full precision kernels, and (c) binarization after training (applying DCQ). In the figure, weights that are different between the trained binary kernel and the directly binarized are highlighted with square rectangles across the three visualizations. Spatially contrasting those highlighted altered weights on the full precision kernels, it can be noticed that they mostly share a common feature that is being low in magnitude (shown as near white squares in (a)). We can observe the following. First, during training, only very small percentage of the weights impacted by training and are actually altered relative to the total number of weights. Moreover, despite the marginal difference between the binary kernels, they experience dramatic accuracy difference: 10.8% vs 98.8% for kernels in (b) and (c) respectively. From statistical

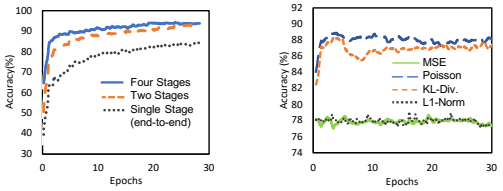

Figure 5. (a) Impact of different splitting on the convergence behavior, (b) Impact of loss formulations on the convergence behavior.

point of view, Figure 4 shows the original full-precision weights histogram (in blue) and overlaying the portion of the altered weights (in light orange) for more networks: VGG-11, and ResNet-20. As seen in the figure, the altered portion of DCQ binary weights is consistently smaller in number and magnitude across different layers and networks. Comparing the two training algorithms, *DCQ yields minimal changes in the right place* to the binary weights as the entire technique is based on matching the intermediate features represented by weight kernels. Which, consequently yields faster convergence and higher accuracies at the same time.

Impact of the number of splitting points. As number of splitting points increases, the large optimization problem gets divided into smaller subproblems; thus, making each subproblem easier to solve separately. Figure 5 (a) shows the convergence behavior for different splittings of VGG-11: four-stage and two-stage splitting as compared to single stage knowledge distillation. As seen in the figure, not only the convergence is faster as number of stages increases but it also eventually converges to a higher final accuracy as compared to lesser number of stages or no splitting at all.

Impact of different loss formulations for intermediate learning. We have examined various loss formulations for intermediate features learning: Poisson, KL-divergence, Mean Squared Error (MSE), and L1-Loss. Figure 5 (b) depicts the convergence behavior for these four loss formulations. As both Poisson Loss and KL-Divergence are a measure of how far two distributions are from each other, they seem to have better performance than the traditional loss functions.

5. Conclusion

Quantization offers a promising path forward to reduce the compute complexity and memory footprint of deep neural networks. To that end, we developed a sectional multi-backpropagation algorithm that leverages multiple instances of knowledge distillation and intermediate feature representations to train a quantized network through divide and conquer. This algorithm, DCQ, achieves significant accuracy improvement to the state-of-the-art quantization methods by exploring a new sectional approach towards knowledge distillation.

References

Bucila, C., Caruana, R., and Niculescu-Mizil, A. Model compression. In Eliassi-Rad, T., Ungar, L. H., Craven, M., and Gunopulos, D. (eds.), *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, pp. 535–541. ACM, 2006. ISBN 1-59593-339-5. doi: 10.1145/1150402.1150464. URL https://doi.org/10.1145/1150402.1150464.

Choi, J., Wang, Z., Venkataramani, S., Chuang, P. I.-J., Srinivasan, V., and Gopalakrishnan, K. Pact: Parameterized clipping activation for quantized neural networks. *CoRR*, abs/1805.06085, 2018.

Courbariaux, M., Bengio, Y., and David, J. Binaryconnect: Training deep neural networks with binary weights during propagations. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 3123–3131, 2015.

Gupta, S., Agrawal, A., Gopalakrishnan, K., and Narayanan, P. Deep learning with limited numerical precision. In Bach, F. R. and Blei, D. M. (eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 1737–1746. JMLR.org, 2015. URL http://jmlr.org/proceedings/papers/v37/gupta15.html.

Hauswald, J., Laurenzano, M., Zhang, Y., Li, C., Rovinski, A., Khurana, A., Dreslinski, R. G., Mudge, T. N., Petrucci, V., Tang, L., and Mars, J. Sirius: An open end-to-end voice and vision personal assistant and its implications for future warehouse scale computers. In *ASPLOS*, 2015.

Hinton, G. E., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. URL http://arxiv.org/abs/1503.02531.

Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., and Bengio, Y. Quantized neural networks: Training neural networks with low precision weights and activations. *Journal of Machine Learning Research*, 18:187:1–187:30, 2017a. URL http://jmlr.org/papers/v18/16-456.html.

Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., and Bengio, Y. Quantized Neural Networks: Training Neural Networks with Low Precision Weights and Activations. *J. Mach. Learn. Res.*, 2017b.

Jaderberg, M., Czarnecki, W. M., Osindero, S., Vinyals, O., Graves, A., Silver, D., and Kavukcuoglu, K. Decoupled neural interfaces using synthetic gradients. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1627–1635. PMLR, 2017. URL http://proceedings.mlr.press/v70/jaderberg17a.html.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In Bartlett, P. L., Pereira, F. C. N., Burges, C. J. C., Bottou, L., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, pp. 1106–1114, 2012.

LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., and Jackel, L. D. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1:541–551, 1989.

LeCun, Y., Bengio, Y., and Hinton, G. E. Deep learning. *Nature*, 521(7553):436–444, 2015. doi: 10.1038/nature14539. URL https://doi.org/10.1038/nature14539.

Li, F. and Liu, B. Ternary Weight Networks. *CoRR*, abs/1605.04711, 2016.

Mishra, A. and Marr, D. Apprentice: Using Knowledge Distillation Techniques To Improve Low-Precision Network Accuracy. In *International Conference on Learning Representations*, 2018.

Mishra, A. K., Nurvitadhi, E., Cook, J. J., and Marr, D. WRPN: Wide Reduced-Precision Networks. In *ICLR*, 2018.

Rastegari, M., Ordonez, V., Redmon, J., and Farhadi, A. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks. In *ECCV*, 2016.

Romero, A., Ballas, N., Kahou, S. E., Chassang, A., Gatta, C., and Bengio, Y. Fitnets: Hints for thin deep nets. In Bengio, Y. and LeCun, Y. (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6550.

Zhou, A., Yao, A., Guo, Y., Xu, L., and Chen, Y. Incremental network quantization: Towards lossless cnns with low-precision weights. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=HyQJ-mclg.

Zhou, S., Ni, Z., Zhou, X., Wen, H., Wu, Y., and Zou, Y. DoReFa-Net: Training Low Bitwidth Convolutional Neural Networks with Low Bitwidth Gradients. *CoRR*, 2016.

Zhu, C., Han, S., Mao, H., and Dally, W. J. Trained Ternary Quantization. In *ICLR*, 2017.

Zmora, N., Jacob, G., and Novik, G. Neural network distiller, June 2018. URL https://doi.org/10.5281/zenodo.1297430.