AN ANALYTICAL METHOD TO DETERMINE MINIMUM PER-LAYER PRECISION OF DEEP NEURAL NETWORKS

Charbel Sakr and Naresh Shanbhag

Dept. of Electrical and Computer Engineering, University of Illinois at Urbana Champaign

ABSTRACT

There has been growing interest in the deployment of deep learning systems onto resource-constrained platforms for fast and efficient inference. However, typical models are overwhelmingly complex, making such integration very challenging and requiring compression mechanisms such as reduced precision. We present a layer-wise granular precision analysis which allows us to efficiently quantize pre-trained deep neural networks at minimal cost in terms of accuracy degradation. Our results are consistent with recent findings that perturbations in earlier layers are most destructive and hence needing more precision than in later layers. Our approach allows for significant complexity reduction demonstrated by numerical results on the MNIST and CIFAR-10 datasets. Indeed, for an equivalent level of accuracy, our fine-grained approach reduces the minimum precision in the network up to 8 bits over a naive uniform assignment. Furthermore, we match the accuracy level of a state-of-the-art binary network while requiring up to $\sim 3.5\times$ lower complexity. Similarly, when compared to a state-of-the-art fixed-point network, the complexity savings are even higher (up to $\sim 14\times$) with no loss in accuracy.

Index Terms— deep learning, neural networks, precision, analysis

1. INTRODUCTION

It is well appreciated that neural networks are very powerful predictive models. However, most state-of-the-art neural networks are overwhelmingly complex. It is common to find networks requiring around 1 billion multiply-accumulates (MACs) [1], or having over 100 million parameters [2] and over 1000 layers [3]. Such high complexity has motivated researchers to find ways of reducing the resource utilization of neural networks. Among recent approaches are pruning [4], sequential reduction [5], and zero skipping [1].

It was shown to be possible to directly train 16-bit fixed-point networks using stochastic rounding [6], or even fully binarized networks [7, 8, 9]. Theoretically, undertaking such a discrete optimization is significantly more challenging than the usual, full precision, back-propagation based training of neural networks. Furthermore, the inherent robustness of neural networks suggests that quantizing a pre-trained model would provide a much easier, yet reliable, way of reducing the complexity. This has led to some interesting analytical investigations on the quantization tolerance of neural networks. In [10], an analytical solution for customized fixed-point representations based on the signal-to-quantization-noise ratio (SQNR) was presented. However, this solution suffers from the SQNR not being a meaningful metric in the context of classification and machine learning. An elegant solution has been recently provided in [11] where analytical upper bounds on the accuracy degradation as a function of precision were derived. In addition, separate precisions were assigned for activations and weights, and led to the discovery of an interesting trade-off between the two.

Recently [12], it has been empirically observed that the earlier layers in a deep neural network (DNN) can accommodate less perturbations than later (deeper) layers thereby implying that more precision is required by the earlier layers. However, thus far there has been no attempt to exploit this observation and to systematically minimize per-layer precision in DNNs.

In this paper, we employ the DNN precision analysis framework in [11] to provide a theoretical basis and an analytical method for *per-layer precision assignment* in DNNs. The proposed method can be employed by DNN designers to minimize overall precision without having to resort to expensive trial-and-error simulation-based approaches that are prevalent today. We show that per-layer minimum precision of input to output layers varies from 7 bits-to-2 bits and from 11 bits-to-2 bits for networks processing the MNIST [13] and CIFAR-10 [14] datasets, respectively. Therefore, per layer precision assignment leads to much greater savings in complexity compared to a uniform assignment of layer precisions. Indeed, for the MNIST and CIFAR-10 datasets, and for the same level of accuracy, we show up to 4 bits reduction in minimum precision compared to the coarse-grained approach of [11] and up to 8 bits reduction compared to a naive uniform assignment [6, 1]. Moreover, we achieve same accuracy but $\sim 3.5\times$ less complexity than a state-of-the-art binary network, BinaryNet [7]. Compared to a state-of-the-art fixed-point network [6], the complexity savings are even higher (up to $\sim 14\times$) in spite of better accuracy.

The rest of this paper is organized as follows. Section 2 presents necessary background. Section 3 introduces our proposed precision reduction method. Numerical results are included in Section 4. We conclude our paper in Section 5.

2. BACKGROUND

We first review the theoretical results of [11] relating accuracy to precision. The accuracy metric used is the mismatch

This work was supported in part by Systems On Nanoscale Information fabriCs (SONIC), one of the six SRC STARnet Centers, sponsored by MARCO and DARPA.

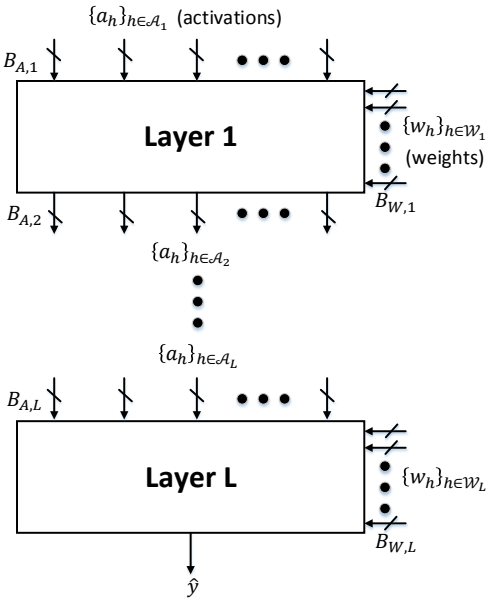

Fig. 1: Per-layer precision assignments for a feedforward neural network architecture. Each layer performs either a matrix vector multiplication, or a set of 2D convolutions, followed by a non-linear activation function. The precision assignments for layer l are $B_{A,l}$ and $B_{W,l}$ for activations and weights, respectively.

probability $p_m = P\left(\hat{Y}_{fx} \neq \hat{Y}_{fl}\right)$, which is the probability that the predicted label $\hat{Y}_{fx}$ of a fixed-point neural network is different from $\hat{Y}_{fl}$, that of its floating point counterpart. It was shown [11] that:

$$p_m \leq \Delta_A^2 E_A + \Delta_W^2 E_W \tag{1}$$

where $\Delta_A = 2^{-(B_A-1)}$ and $\Delta_W = 2^{-(B_W-1)}$ are the activation and weight quantization step-sizes, respectively, and

$$E_A = \mathbb{E}\left[\sum_{\substack{i=1\\ i\neq\hat{Y}_{fl}}}^{M} \frac{\sum_{h\in\mathcal{A}}\left|\frac{\partial(Z_i - Z_{\hat{Y}_{fl}})}{\partial A_h}\right|^2}{24|Z_i - Z_{\hat{Y}_{fl}}|^2}\right]$$

and

$$E_W = \mathbb{E}\left[\sum_{\substack{i=1\\ i\neq\hat{Y}_{fl}}}^{M} \frac{\sum_{h\in\mathcal{W}}\left|\frac{\partial(Z_i - Z_{\hat{Y}_{fl}})}{\partial w_h}\right|^2}{24|Z_i - Z_{\hat{Y}_{fl}}|^2}\right]$$

are the activation and weight quantization noise gains, respectively. Note the dependence on the number of classes (M), the indexing sets of activations ($\mathcal{A}$) and weights ($\mathcal{W}$), and the soft outputs ($\{Z_i\}_{i=1}^M$).

Interestingly, the noise gains can be obtained as part of a standard back-propagation procedure and need to be computed only once making (1) very practical. Finally, in order to avoid over quantization of either activations or weights, the difference between the two precisions (B_A and B_W) is chosen to balance the sum in (1), as follows:

$$B_A - B_W = round\left(\log_2\sqrt{\frac{E_A}{E_W}}\right) \tag{2}$$

where $round()$ denotes the rounding operation.

Both (1) and (2) can be combined in order to efficiently determine the *coarse-grained* minimum precision requirements of a neural network. Indeed, (2) can first be used to set the difference between B_A and B_W and reduce the search from a 2 dimensional grid to one. Then, (1) can be used to obtain an initial estimate of the precision requirements satisfying a certain maximal condition on p_m. Finally, the search can be fine-tuned beyond the original estimate of (1) via simulations.

3. PROPOSED PRECISION REDUCTION METHOD

3.1. Fine-grained precision analysis

For a given neural network with L layers, let $\{\mathcal{A}_l\}_{l=1}^L$ $\{\mathcal{W}_l\}_{l=1}^L$ be the layer-wise partitions of $\mathcal{A}$ and $\mathcal{W}$, respectively. Consequently, as shown in Fig. 1, if the per-layer precisions are $\{B_{A,l}\}_{l=1}^L$ and $\{B_{W,l}\}_{l=1}^L$ for activations and weights, respectively, then (1) can be re-written as:

$$p_m \leq \sum_{l=1}^{L}\left(\Delta_{A,l}^2 E_{A,l} + \Delta_{W,l}^2 E_{W,l}\right) \tag{3}$$

where $\Delta_{A,l} = 2^{-(B_{A,l}-1)}$ and $\Delta_{W,l} = 2^{-(B_{W,l}-1)}$ are the activation and weight quantization step-sizes at layer l, respectively, and

$$E_{A,l} = \mathbb{E}\left[\sum_{\substack{i=1\\ i\neq\hat{Y}_{fl}}}^{M} \frac{\sum_{h\in\mathcal{A}_l}\left|\frac{\partial(Z_i - Z_{\hat{Y}_{fl}})}{\partial A_h}\right|^2}{24|Z_i - Z_{\hat{Y}_{fl}}|^2}\right]$$

and

$$E_{W,l} = \mathbb{E}\left[\sum_{\substack{i=1\\ i\neq\hat{Y}_{fl}}}^{M} \frac{\sum_{h\in\mathcal{W}_l}\left|\frac{\partial(Z_i - Z_{\hat{Y}_{fl}})}{\partial w_h}\right|^2}{24|Z_i - Z_{\hat{Y}_{fl}}|^2}\right]$$

are the activation and weight quantization noise gains at layer l, respectively.

Observe that (3) is a sum of $2L$ terms where the quantization noise gains are computed only once after training. The design parameters are the $2L$ precision assignments, $\{B_{A,l}\}_{l=1}^L$ and $\{B_{W,l}\}_{l=1}^L$. Once again, a sum of independent terms is to be balanced. To do so, the minimum quantization noise gain is first computed:

$$E_{\min} = \min\left(\{E_{A,l}\}_{l=1}^L, \{E_{W,l}\}_{l=1}^L\right). \tag{4}$$

Then, a reference minimum precision $B_{\min}$ is chosen, and for each layer l, similar to (2), the precision is set as follows:

$$B_{A,l} = round\left(\log_2\left(\sqrt{\frac{E_{A,l}}{E_{\min}}}\right)\right) + B_{\min} \tag{5}$$

and

$$B_{W,l} = round\left(\log_2\left(\sqrt{\frac{E_{W,l}}{E_{\min}}}\right)\right) + B_{\min} \qquad (6)$$

Note that at least one of the 2L precision assignments will equal $B_{\min}$.

Once again, (3)-(6) can be used to efficiently find the *fine-grained* minimum precision requirements of a network. Here, (4), (5), and (6) are used to reduce the search space from a $2L$ dimensional grid to just a one dimensional axis. Afterwards, (3) is used to provide an initial estimate of precision requirements. Note that the search space reduction is massive. For instance, for a 5 layer network, if we are considering precisions up to 16 bits, the search space is reduced from 16^{10} to only 16 design points.

3.2. Complexity in Fixed-point

In order to quantify the benefits of the proposed precision reduction method, we shall consider two measures of complexity [11]: *computational* and *representational* costs.

The computational cost is the total number of full adders needed per decision. Note that each layer implements an ensemble of dot products in order to realize either a matrix vector multiplication or a set of 2D convolutions. Hence, the computational cost (measured in 1 bit full adders or FAs) of a network is [11]:

$$\sum_{l=1}^{L}\Big[N_l\left(D_l\, B_{A,l} B_{W,l} + \right. \\ \left. (D_l - 1)(B_{A,l} + B_{W,l} + \lceil \log_2(D_l)\rceil - 1)\right)\Big]$$

where N_l and D_l are the number and dimensionality of dot products computed at layer l, respectively.

The representational cost is the total number of bits needed to represent both weights and activations, and is given by:

$$\sum_{l=1}^{L}\left(|\mathcal{A}_l|\, B_{A,l} + |\mathcal{W}_l|\, B_{W,l}\right)$$

4. NUMERICAL RESULTS

4.1. Set-up

Numerical experiments on two datasets are considered: the MNIST dataset for character recognition [13], and the CIFAR-10 dataset for object recognition [14]. For each, a neural network is first pre-trained as follows:

- MNIST: A multi-layer perceptron with architecture $784 - 512 - 512 - 512 - 10$. The network is pre-trained using Vanilla SGD [15] and has a baseline test error of 1.10% in floating-point.
- CIFAR-10: A convolutional neural network with architecture $32C3 - 32C3 - MP2 - 64C3 - 64C3 - MP2 - 128C3 - 128C3 - 256FC - 256FC - 10$. The network is pre-trained is trained using Vanilla Adam [16] and has a baseline test error of 11.87% in floating-point.

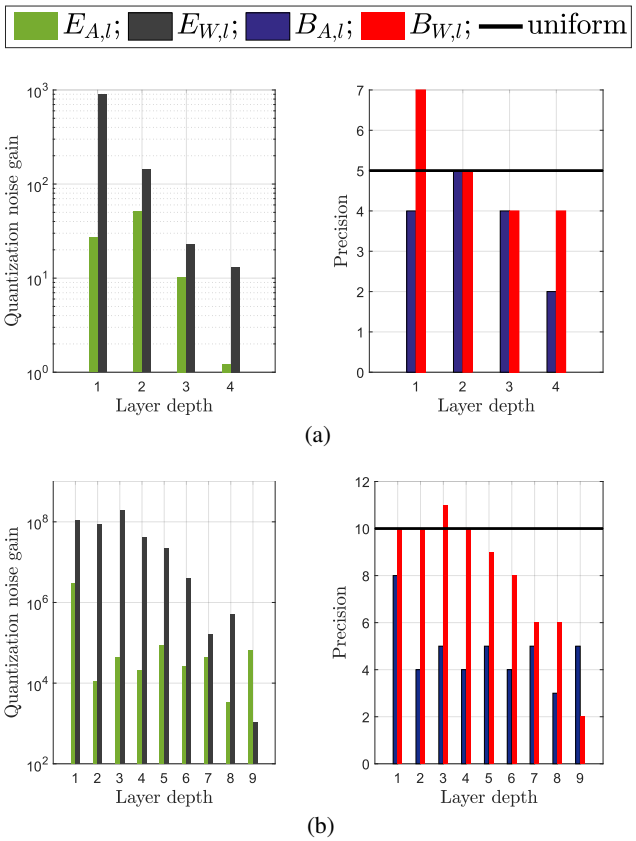

Fig. 2: Plots showing quantization noise gains and per-layer precision assignments for (a) MNIST and (b) CIFAR-10, to satisfy $p_m \leq 1\%$. The minimum precision to satisfy $p_m \leq 1\%$ with uniform precision assignment is also shown.

Note that the architectures described above are inspired from those used by BinaryNet [7] and are actually obtained by reducing the height of each layer by a factor of 4.

Each network is then quantized to fixed-point using three precision assignment methods:

- the proposed *fine-grained* precision assignment.
- a *coarse-grained* precision assignment [11].
- a *uniform* or identical precision assignment for all activations and weights.

The obtained results are compared to those reported by two state-of-the-art works on reduced precision neural networks:

- Stochastic quantization (SQ) [6] - trained fixed-point networks.
- BinaryNet (BN) [7] - trained binarized networks.

4.2. Results

Figure 2 shows the benefits of the proposed approach (3)-(6) compared to a naive uniform precision assignment. Indeed, to satisfy a mismatch probability of $p_m \leq 1\%$, most precisions are less than the required uniform precision. Furthermore, observe that the precision assignment matches the quantization noise gain profile on a logarithmic scale. This is due to the use of (4), (5), and (6). In addition, a general trend of decreasing precision requirements with layer depth is noticed.

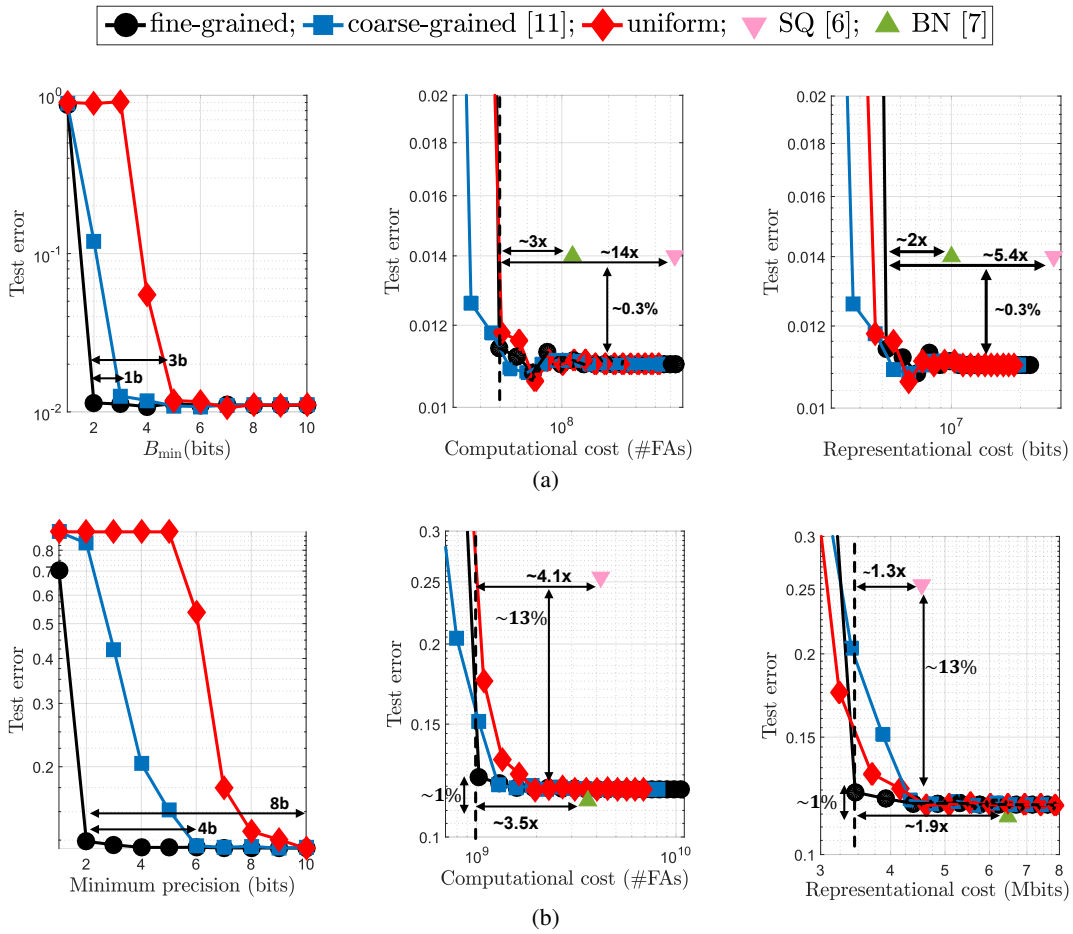

Fig. 3: Test error vs. minimum precision ($B_{\min}$), computational and representational costs, for: (a) MNIST and (b) CIFAR-10 networks. A comparison with SQ [6] and BN [7] is also included.

This is in accordance with recent findings demonstrating that perturbations at the early layers of neural networks are often the most destructive [12]. Here, the proposed method is naturally countering this effect by assigning more precision to the lower levels. Similarly, it is seen that the precision assignments of weights are typically more than those of activations, confirming the findings of [11].

Figures 3 shows that the coarse-grained approach [11] achieves a good initial improvement over the naive uniform assignment, and is able to reduce the minimum precision by up to 4 bits before the accuracy starts to degrade. However, the proposed fine-grained method is noticeably superior and able to reduce the minimum precision to just 2 bits without any notable accuracy degradation for both networks. This corresponds to 4 bits less than the minimum precision obtained via the coarse-grained method for the CIFAR-10 network.

As far as complexity is concerned, the results obtained outperform those of SQ. For instance, on the CIFAR-10 dataset, the test error obtained via the fine-grained precision assignment is $\sim 13\%$ less than that reported by SQ in spite of requiring $\sim 4\times$ fewer computational cost. Moreover, the complexity savings of BN (binarized) are surpassed. Indeed, even though the reported levels of accuracy are very close those obtained via the proposed method, the latter achieves up to $\sim 3\times$ and $\sim 3.5\times$ less computational cost for the MNIST and CIFAR-10 networks, respectively, over the fully binarized BN. Similar trends also hold for the representational costs. These results reflect the importance of depth vs. width vs. precision considerations when exploring reduced complexity neural networks.

5. CONCLUSION

We have presented an analytical approach to fine-grained precision assignment in deep neural networks (DNNs). The benefits of the proposed approach in terms of minimum precision and complexity reduction has been shown. A performance comparison with state-of-the-art binary and fixed-point neural networks was illustrated and highlighted considerable savings. The presented work allows DNN designers to determine minimum precision requirements in DNNs and estimate their complexities without needing to run lengthy simulations. The proposed method can be employed to efficiently explore other dimensions in the design of low-complexity DNNs such as the trade-off between precision vs. depth vs. width, and between precision and pruning.

6. REFERENCES

[1] Yu-Hsin Chen, Tushar Krishna, Joel S Emer, and Vivienne Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, 2017.

[2] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1701–1708.

[3] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger, "Deep networks with stochastic depth," in *European Conference on Computer Vision*. Springer, 2016, pp. 646–661.

[4] Song Han, Jeff Pool, John Tran, and William Dally, "Learning both weights and connections for efficient neural network," in *Advances in Neural Information Processing Systems (NIPS)*, 2015, pp. 1135–1143.

[5] Xiangyu Zhang, Jianhua Zou, Kaiming He, and Jian Sun, "Accelerating very deep convolutional networks for classification and detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015.

[6] S. Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan, "Deep learning with limited numerical precision," in *Proceedings of The 32nd International Conference on Machine Learning*, 2015, pp. 1737–1746.

[7] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio, "Binarized neural networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 4107–4115.

[8] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi, "XNOR-Net: Imagenet classification using binary convolutional neural networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 525–542.

[9] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou, "DoReFa-Net: Training low bitwidth convolutional neural networks with low bitwidth gradients," *arXiv preprint arXiv:1606.06160*, 2016.

[10] Darryl Lin, Sachin Talathi, and Sreekanth Annapureddy, "Fixed point quantization of deep convolutional networks," in *Proceedings of The 33rd International Conference on Machine Learning*, 2016, pp. 2849–2858.

[11] Charbel Sakr, Yongjune Kim, and Naresh Shanbhag, "Analytical guarantees on numerical precision of deep neural networks," in *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 3007–3016.

[12] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein, "On the expressive power of deep neural networks," in *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 2847–2854.

[13] Yann LeCun, Corinna Cortes, and Christopher JC Burges, "The MNIST database of handwritten digits," 1998.

[14] Alex Krizhevsky and Geoffrey Hinton, "Learning multiple layers of features from tiny images," 2009.

[15] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*, pp. 177–186. Springer, 2010.

[16] Diederik Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.