SNDCNN: SELF-NORMALIZING DEEP CNNs WITH SCALED EXPONENTIAL LINEAR UNITS FOR SPEECH RECOGNITION

Zhen Huang, Tim Ng, Leo Liu, Henry Mason, Xiaodan Zhuang, Daben Liu

Apple Inc., USA

{zhen_huang,tim_ng,lliu9,hmason,xiaodan_zhuang,daben_liu}@apple.com

ABSTRACT

Very deep CNNs achieve state-of-the-art results in both computer vision and speech recognition, but are difficult to train. The most popular way to train very deep CNNs is to use shortcut connections (SC) together with batch normalization (BN). Inspired by Self-Normalizing Neural Networks, we propose the self-normalizing deep CNN (SNDCNN) based acoustic model topology, by removing the SC/BN and replacing the typical RELU activations with scaled exponential linear unit (SELU) in ResNet-50. SELU activations make the network self-normalizing and remove the need for both shortcut connections and batch normalization. Compared to ResNet-50, we can achieve the same or lower (up to 4.5% relative) word error rate (WER) while boosting both training and inference speed by 60%-80%. We also explore other model inference optimization schemes to further reduce latency for production use.

Index Terms: shortcut connection, batch normalization, scaled exponential linear units, self-normalization, ResNet, very deep CNNs

1. INTRODUCTION

Very deep CNNs achieve state-of-the-art results on various tasks [1] in computer vision. Network depth has been crucial in obtaining those leading results [1, 2]. Naïve deep stacking of layers typically leads to a vanishing/exploding gradients problem, making convergence difficult or impossible. For example, VGGNet [1] only uses 18 layers. Normalization methods, including batch normalization [3], layer normalization [4] and weight normalization [5], allow deeper neural nets to be trained. Unfortunately, these normalization methods make training stability sensitive to other factors, such as SGD, dropout, and the estimation of normalization parameters. Accuracy often saturates and degrades as network depth increases [6, 7].

ResNet [8] uses shortcut connections (SC) and batch normalization (BN), allowing the training of surprisingly deep architectures with dramatic accuracy improvements. Since its invention, ResNet has dominated the field of computer vision. The later state-of-the-art-model, DenseNet [9], also uses SC and BN. Besides success in computer vision, ResNet has also performed well in acoustic models for speech recognition [10, 11].

An alternative solution to the problem of vanishing/exploding gradients is self-normalizing neural networks[12]. SNNs use the scaled exponential linear unit (SELU) activation function to induce self-normalization. SNNs have been shown to converge very deep networks without shortcut connections or batch normalization. SNNs are also robust to perturbations caused by training regularization techniques.

Very deep convolutional neural network acoustic models are computationally expensive when used for speech recognition. Several techniques have been explored to improve inference speed on commodity server CPUs. Batching and lazy evaluation have been shown to improve inference speed on CPUs [13] for neural networks of all types. Specifically for speech recognition, running inference at a decreased frame rate [14] has also been shown to reduce computation cost without affecting accuracy too much. We use frame-skipping and multi-threaded lazy computation.

Inspired by [12], we propose another way to train very deep networks without SC and BN by utilizing SELU activations. Experimental results in speech recognition tasks show that by removing the SC/BN and replacing the RELU activations with SELU in ResNet50, we can always get lower WER (up to 4.5% relative) than ResNet50 and 60%-80% training and inference speedup. We further speech up the decoding by applying techniques such as frame skipping and multi-thread lazy computation.

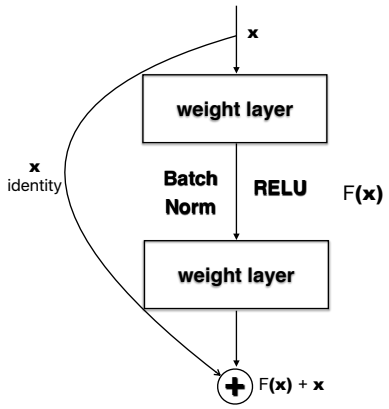

Fig. 1. Typical building block of ResNet

2. RELATED WORK

2.1. Residual Learning

ResNet [8] solves many problems in training very deep CNNs. The key ResNet innovation is the shortcut connections shown in Figure 1 which depicts a typical building block of ResNet. The input to the block, x, will go through both the original mapping $F(x)$ (weight layers, RELU activations and batch normalization [3]) and the identity shortcut connection. The output, y, will be $F(x) + x$. The authors in [8] hypothesize that the so-called residual mapping of $y = F(x)+x$ should be easier to optimize than the original mapping of $y = F(x)$. The design of the special building block is motivated by the observation in [6, 7] that accuracy degrades when more layers

are stacked onto an already very deep CNN model. If the added layers can be constructed as identity mappings, the deeper model should not have worse training error than the original shallower model without these added layers. The degradation actually suggests that the optimizer has difficulties in approximating identity mappings. With the identity shortcut connections in the ResNet building block, the optimizer can simply drive the layer weights toward zero to make the block identity mapping. ResNet-style CNNs have maintained state-of-the-art results and have inspired other model structures [9, 15].

2.2. Batch Normalization

Besides the shortcut connections shown in Figure 1, batch normalization (BN) [3] is also an important feature of ResNet. BN is designed to reduce internal covariate shift, defined as the change in the distribution of network activations due to the change in network parameters, during training. This ensures better and faster convergence of the training process. BN is achieved by whitening the input of each layer, but full whitening of each layer's inputs is costly and not differentiable everywhere. Instead of whitening the features in layer inputs and outputs jointly, each scalar feature is normalized independently to zero mean and unit variance. For a layer with d-dimensional input $x = (x(1)...x(d))$, each dimension will be normalized as:

$$\widehat{x}^{(k)} = \frac{x^{(k)} - \mathbf{E}[x^{(k)}]}{\sqrt{\mathbf{Var}[x^{(k)}]}} \tag{1}$$

BN also ensures that the normalization can represent the identity transform by introducing a pair of parameters $\gamma^{(k)}$, $\beta^{(k)}$, which scale and shift the normalized value $\widehat{x}^{(k)}$:

$$y^{(k)} = \gamma^{(k)}\widehat{x}^{(k)} + \beta^{(k)}. \tag{2}$$

In mini-batch based stochastic optimization, the mean $\mathbf{E}[x^{(k)}]$ and variance $\mathbf{Var}[x^{(k)}]$ are estimated within each mini-batch.

BN has been successfully adopted in various tasks, but training with BN can be perturbed by many factors such as SGD, dropout, and the estimation of normalization parameters. Moreover, in order to fully utilize BN, samples in each mini-batch must be i.i.d [16]. However, state-of-the-art speech recognition requires sequence level training of the acoustic model [17]. In sequence level training, a mini-batch consists of all the frames of a single utterance, and the frames are highly correlated to each other. This violates the i.i.d requirement of BN, making batch normalization very challenging to use with sequence training.

2.3. Self-Normalizing Neural Networks

[12] introduces self-normalizing neural networks (SNNs) in which neuron activations automatically converge towards zero mean and unit variance. The key to inducing the self-normalizing properties in SNNs is the special activation function, the scaled exponential linear unit (SELU), formulated as:

$$\mathbf{selu}(x) = \lambda \begin{cases} x & \text{if } x > 0 \\ \alpha e^x - \alpha & \text{if } x \leq 0 \end{cases} \tag{3}$$

with $\alpha \approx 1.6733$ and $\lambda \approx 1.0507$.

The values of α and λ are obtained by solving fixed point equations to give the activation function the following characteristics, which ensures the self-normalizing property [12]:

1 Negative and positive values for controlling the mean

2 Saturation regions (derivatives approaching zero) to dampen the variance if it is too large in the lower layer

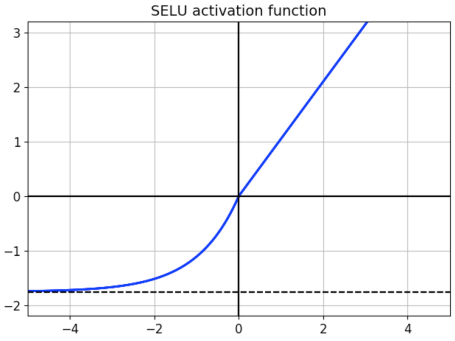

Fig. 2. SELU activation function

3 A slope larger than one to increase the variance if it is too small in the lower layer

4 A continuous curve

The shape of SELU activation function is shown in Figure 2. Using SELU, SNNs push neuron activations to zero mean and unit variance. This gives us the same effect as batch normalization without being prone to the perturbations discussed in Section 2.2.

3. TRAINING SELF-NORMALIZING VERY DEEP CNNS

We revise the model topology discussed in [8] and design the proposed Self-Normalizing Deep CNNs (SNDCNN) for a hybrid automatic speech recognition system [18]. The building block for SNDCNN is shown in Figure 3. Comparing Figure 1 and 3, we can see that the shortcuts and batch normalization are removed, and the activation function is changed to SELU. We thus practically obtain a self-normalizing ResNet.

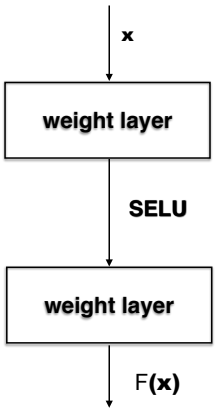

Fig. 3. Building block of SNDCNN

We verify the Self-Normalizing property by observing the trend of mean and variance in the SELU activation outputs during training. The model topology is a 50-layer CNN obtained by removing SC and BN from ResNet-50. We call this topology SNDCNN-50. Model parameters are initialized as instructed in [12]. In Figures 4 and 5,

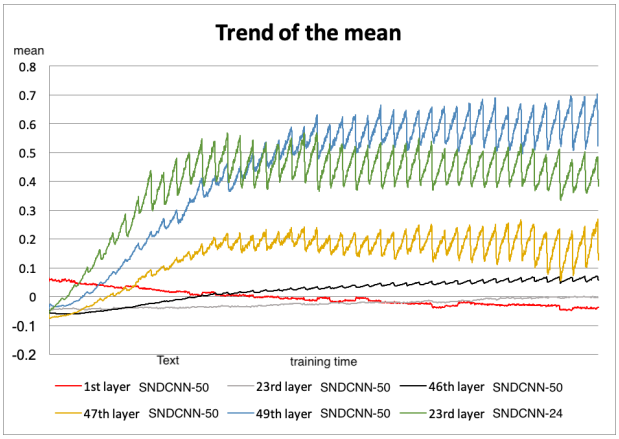

Fig. 4. Trend of the mean

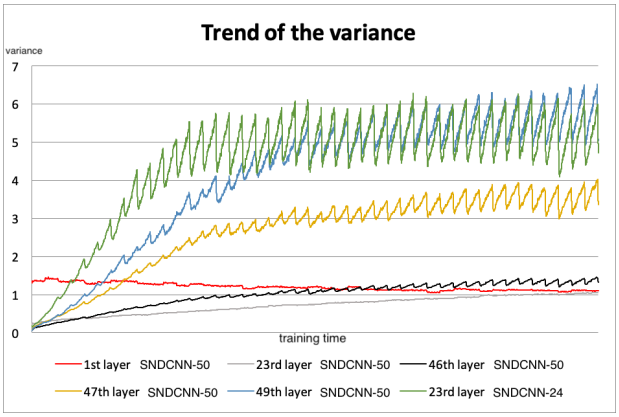

Fig. 5. Trend of the variance

we plot the mean and variance trend of the 1st, 23rd, 46th, 47th, and 49th layers of SNDCNN-50 and the 23rd layer of SNDCNN-24. The mean and variance are computed across frames within a mini-batch (256 frames). Each data point is obtained by averaging all the units in the same layer. The x-axis is training time, and we collect statistics from 33k mini-batches to draw each curve.

In the SNDCNN-50 case, we can see that the outputs of 1st and middle (23rd) layers follow the claims in [12] nicely, but the last several layers do not. We find that the non-self-normalizing phenomenon becomes significant only after the 46th layer. As shown in Figure 4 and 5, the 46th layer almost has mean $= 0$ and variance $= 1$, but the following layers are worse. We verify that the non-self-normalizing phenomenon is not caused by the depth of the neural network but by the distance to the output layer. The 23rd layer of SNDCNN-24 has the non-self-normalizing phenomenon, similar to the one seen in the 49th layer of SNDCNN-50, while the 23rd layer of SNDCNN-50 has a very nice self-normalizing property. We suspect that the back propagation path has to be long enough to effectively train the neural network's parameters to ensure the self-normalizing property. Although the last layers do not strictly follow [12]'s self-normalizing claim, the mean and variance are reasonable (mean < 0.8, variance < 9) even after 109 million mini-batches (28 billion training samples). We also evaluated different kinds of initialization for the network. Our findings indicate that as long as training starts normally, the trend of the mean and variance will follow the patterns seen in Figures 4 and 5.

Removing SC and BN simplifies the model structure and speeds up both training and inference. Removing BN also solves the sequence level training problem discussed in Section 2.2. Most importantly, we always observe as good or better accuracy with the proposed simplified model structure.

Table 1. WERs (in %) of different model topologies with 300h training and 7h testing data in en_US

0	Model	WER
1	6 layer DNN w/ RELU	16.2%
2	6 layer DNN w/ SELU	16.0%
3	30 layer DNN w/ RELU	not trainable
4	30 layer DNN w/ SELU	15.9%
5	ResNet-50 w/RELU w/ SC&BN (standard ResNet)	15.3%
6	ResNet-50 w/SELU w/ SC&BN	15.2%
7	ResNet-50 w/RELU w/o SC&BN	not trainable
8	ResNet-50 w/SELU w/o SC&BN (SNDCNN-50)	14.9%

Table 2. CERs (in %) of different model topologies with 4000h training and 30h testing data in zh_CN

0	Model	WER
1	ResNet-50 w/RELU w/ SC&BN (standard ResNet)	8.8%
2	ResNet-50 w/RELU w/o SC w/ BN	8.9%
3	ResNet-50 w/RELU w SC w/o BN	8.7%
4	ResNet-50 w/RELU w/o SC&BN	not trainable
5	ResNet-50 w/SELU w/ SC&BN	8.7%
6	ResNet-50 w/SELU w/o SC&BN (SNDCNN-50)	8.7%

4. EXPERIMENTS

All data used in this work comes from Siri internal datasets (en_US and zh_CN). All models are trained with Blockwise Model-Update Filtering (BMUF) [19] with 32 GPUs. Newbob learning scheduling is used for all the experiments. A 4-gram language model is used in all experiments. 40 dimensional filter bank feature is extracted with 25ms window and 10ms step size. All the models use a context window of 41 frames (20-1-20) as the visible states [20].

4.1. Accuracy

Table 1 compares WERs of different model topologies for en_US. The training data contains 300 hours of speech, and the testing data covers 7 hours of speech. From Table 1, we have the following observations:

1 [Row 1-4 vs. Row 5-8] Deep CNN models show advantage in terms of WER against shallower DNNs

2 [Row 3 vs. Row 4] [Row 7 vs. Row 8] SELU activation makes the training of very deep models (with no SC&BN) feasible

3 [Row 1 vs. Row 2] [Row 5 vs. Row 6] SELU activation is no worse than RELU in DNN or ResNet topology.

4 [Row5 vs. Row 8] SNDCNN obtains better WER than ResNet

Table 3. WERs (in %) with different model topologies with 10000h training and 7h testing data in en_US

0	Model	WER
1	ResNet-50	8.8%
2	SNDCNN-50	8.4%

Table 4. Speedups (in %) with different model topologies against standard ResNet-50

0	Model	Training	Inference
1	ResNet-50	0%	0%
2	ResNet-50 w/RELU w/o SC w/ BN	19.4%	30.0%
3	ResNet-50 w/RELU w SC w/o BN	34.6%	49.7%
4	SNDCNN-50	57.8%	80.6%

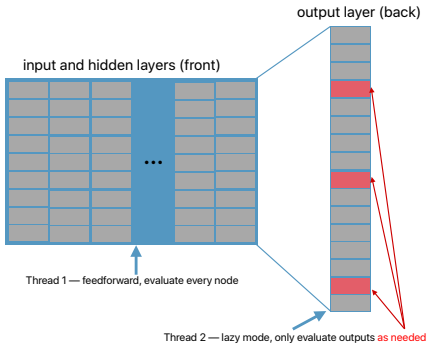

Fig. 6. Multi-threaded lazy evaluation for acoustic model inference

Table 2 compares character error rate (CER) of different model topologies for zh_CN. The training data contains 4000 hours of speech and the testing data consists of 30 hours of speech. From Table 2, we find that in order to make the training of very deep CNNs feasible, we must use at least one of the following three techniques: batch normalization, shortcut connection, and SELU activation. The WERs of different topologies with the same depth are actually very similar. This phenomenon suggests that depth could be the key to better accuracy. The proposed SNDCNN has slightly better WER than ResNet.

Table 3 compares en_US WER of ResNet-50 and SNDCNN-50 with 10000 hours of training data and 7 hours of testing data. In this experiment, the proposed SNDCNN has much better WER than ResNet.

4.2. Speedup

Table 4 shows the relative computation speedups (frames per second) of the variants considered in Table 2. From Table 2, we know that the 4 models in Table 4 have very similar WER. but from Table 4, we can find that removal of BN and SC results in significant speedup in both training and inference. The speedup (especially in inference) is very important in deploying SNDCNN-50 in production systems where minimising latency is essential.

5. INFERENCE PERFORMANCE OPTIMIZATION

We already achieve significant inference speedup by removing BN and SC from ResNet-50 as discussed in Section 4.2. Further inference optimization for SNDCNN-50 was investigated, particularly frame-skipping and multi-threaded lazy computation.

Table 5. Latency reduction (in %) with different inference techniques

0	Technique	Latency reduction
1	Frame-skipping	47.2%
2	Multi-thread lazy mode	10.8%

Frame-skipping [14]: our acoustic model targets tied HMM (hidden Markov model) states [21], running at 100 frames per second, but the predictions do not frequently change between frames. Human speech rarely has more than 10 phonemes per second. By simply skipping and duplicating two thirds of frames, we reduce the required computation by 3x which translates into 47.2% latency reduction as shown in Table 5. Note that usually skipping frames will result in some WER degradation [14] and we indeed observed that in our experiments with shallower models (10 layer, 2 convolution layer plus 8 fully connected) even when we skip only half of the frames. However, with SNDCNN-50, we can skip up to two thirds of frames with no degradation on WER.

Multi-thread lazy computation [13]: as shown in Figure 6, we split the acoustic model into two parts: front and back. We use two threads to do the inference independently. Thread 1 will do the inference of the front part which contains the input and hidden layers. Thread 2 will do the inference of the back part which contains the output layer. The outputs target tied HMM states, and can easily be more than 10 thousand. As performing inference for the entire layer is expensive, we only compute the outputs that are needed by the decoding graph instead of computing every output of the layer. By doing this "lazy" on-demand inference, we save a lot of computation in the large output layer, which translates into a 10.8% latency reduction as shown in Table 5.

6. CONCLUSIONS

In this paper, we proposed a very deep CNN based acoustic model topology SNDCNN, by removing the SC/BN and replacing the typical RELU activations with scaled exponential linear unit (SELU) in ResNet-50. This leverages self-normalizing neural networks, by use of scaled exponential linear unit (SELU) activations, to train very deep convolution networks, instead of residual learning [8]). With the self-normalization ability of the proposed network, we find that the SC and BN are no longer needed. Experimental results in hybrid speech recognition tasks show that by removing the SC/BN and replacing the RELU activations with SELU in ResNet-50, we can achieve the same or lower WER and 60%-80% training and inference speedup. Additional optimizations in inference, specifically frame skipping and lazy computation with multi-threading, further speed up the SNDCNN-50 model by up to 58% which achieves production quality accuracy and latency.

7. ACKNOWLEDGMENTS

The authors would like to thank Professor Steve Young, Bing Zhang, Roger Hsiao, Xiaoqiang Xiao, Chao Weng and Professor Sabato Marco Siniscalchi for valuable discussions and help.

8. REFERENCES

[1] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[2] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. CVPR*, 2015, pp. 1–9.

[3] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015, pp. 448–456.

[4] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv:1607.06450*, 2016.

[5] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in *Proc. NeurIPS*, 2016, pp. 901–909.

[6] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *Proc. CVPR*, 2015, pp. 5353–5360.

[7] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," 2015.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. NeurIPS*, June 2016.

[9] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. CVPR*, 2017, pp. 4700–4708.

[10] G. Saon, H.-K. J. Kuo, S. Rennie, and M. Picheny, "The IBM 2015 english conversational telephone speech recognition system," *arXiv:1505.05899*, 2015.

[11] W. Xiong, L. Wu, F. Alleva, J. Droppo, X. Huang, and A. Stolcke, "The Microsoft 2017 conversational speech recognition system," in *Proc. ICASSP*. IEEE, 2018, pp. 5934–5938.

[12] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," in *Proc. NeurIPS*, 2017, pp. 971–980.

[13] V. Vanhoucke, A. Senior, and M. Z. Mao, "Improving the speed of neural networks on cpus," in *Proc. NeurIPS*, 2011.

[14] V. Vanhoucke, M. Devin, and G. Heigold, "Multiframe deep neural networks for acoustic modeling," in *Proc. ICASSP*, 2013.

[15] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proc. AAAI*, 2017.

[16] S. Ioffe, "Batch renormalization: Towards reducing minibatch dependence in batch-normalized models," in *Proc. NeurIPS*, 2017, pp. 1945–1953.

[17] K. Veselỳ, A. Ghoshal, L. Burget, and D. Povey, "Sequence-discriminative training of deep neural networks." in *Proc. Interspeech*, vol. 2013, 2013, pp. 2345–2349.

[18] H. A. Bourlard and N. Morgan, *Connectionist speech recognition: a hybrid approach*. Springer Science & Business Media, 2012, vol. 247.

[19] K. Chen and Q. Huo, "Scalable training of deep learning machines by incremental block training with intra-block parallel optimization and blockwise model-update filtering," in *Proc. ICASSP*. IEEE, 2016, pp. 5880–5884.

[20] A.-r. Mohamed, G. Dahl, and G. Hinton, "Deep belief networks for phone recognition," in *NeurIPS workshop on deep learning for speech recognition and related applications*, vol. 1, no. 9, 2009, p. 39.

[21] S. J. Young, J. J. Odell, and P. C. Woodland, "Tree-based state tying for high accuracy acoustic modelling," in *Proceedings of the workshop on Human Language Technology*. Association for Computational Linguistics, 1994, pp. 307–312.