RESNET IN RESNET: GENERALIZING RESIDUAL ARCHITECTURES

Sasha Targ[*], **Diogo Almeida**[*], **Kevin Lyman**
Enlitic
sasha.targ@gmail.com,{diogo,kevin}@enlitic.com

ABSTRACT

ResNets have recently achieved state-of-the-art results on challenging computer vision tasks. In this paper, we create a novel architecture that improves ResNets by adding the ability to forget and by making the residuals more expressive, yielding excellent results. ResNet in ResNet outperforms architectures with similar amounts of augmentation on CIFAR-10 and establishes a new state-of-the-art on CIFAR-100.

1 INTRODUCTION

Recently proposed residual networks (ResNets) (He et al., 2015) which make use of shortcut connections similar to those in Highway networks (Srivastava et al., 2015) get state-of-the-art performance on the ILSVRC 2015 classification task and allow training of extremely deep networks up to over 1000 layers. However, the current ResNet has several architectural limitations in that residuals must be learned by fixed size shallow subnetworks, despite evidence that deeper networks are more expressive. Identity connections as implemented in the current ResNet also result in a mix of levels of feature representation at each layer, even though some features learned at earlier layers of a deep network may no longer provide useful information in later layers. A prior of the ResNet architecture is that learning identity weights is difficult, but by the same argument, it is difficult to learn the additive inverse of identity weights to remove information from the representation at any given layer.

We present a novel architecture which resolves the issues outlined above by incorporating residual and non-residual computation streams to generalize both residual networks and standard feedforward neural networks and demonstrate state-of-the-art performance on CIFAR-100 using this architecture.

2 GENERALIZING RESIDUAL NETWORK ARCHITECTURES

Our architecture (see Figure 1b) consists of convolutional layers partitioned into two streams of processing: a residual stream and a working memory stream. The residual stream $\mathbf{r}$ resembles the original structure of a ResNet (He et al., 2015) with identity connections between each unit of processing. The working memory stream $\mathbf{m}$ adds the ability to process information from either stream in a nonlinear manner without identity connections. The output of each stream at each block is a separate convolutional kernel applied to each input stream, with the addition of a shortcut connection from the input to the output of the residual stream (Equation 1). The form of the shortcut connection can be an identity function with the appropriate padding or a projection as in He et al. (2015).

$$\begin{aligned} \mathbf{r}_{l+1} &= \sigma(\text{conv}(\mathbf{r}_l, W_{l,1}) + \text{conv}(\mathbf{m}_l, W_{l,2}) + \text{shortcut}(\mathbf{r}_l)) \\ \mathbf{m}_{l+1} &= \sigma(\text{conv}(\mathbf{r}_l, W_{l,3}) + \text{conv}(\mathbf{m}_l, W_{l,4})) \end{aligned} \tag{1}$$

By repeating this generalized residual block several times, the network architecture has the expressivity to learn either a standard CNN (by zeroing out the residual stream) or a single layer ResNet

[*]Equal contribution. Author ordering determined by coin flip.

(by zeroing out the working memory stream) and anything in between, including the standard two layer ResNet architecture (Figure 1c). The architecture furthermore allows the network to learn residuals with a variable effective number of processing steps before addition back into the residual stream.

This architecture is not specific to CNNs, and can be applied to standard fully connected layers and other feedforward layers.

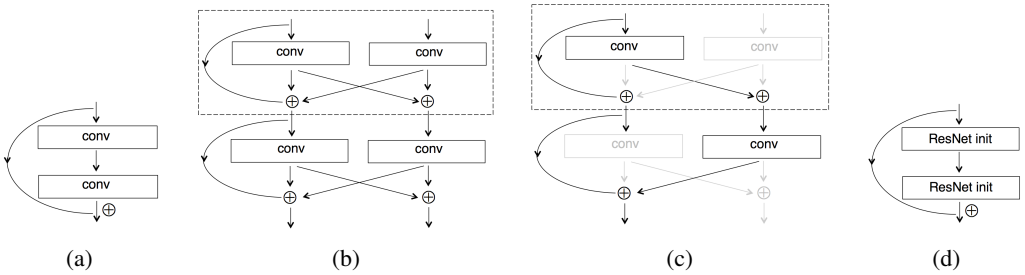

Figure 1: Left: 2-layer ResNet block. Middle-left: 2 generalized ResNet blocks. Middle-right: 2-layer ResNet block from generalized ResNet. Right: 2-layer ResNet in ResNet block.

2.1 RESNET INITIALIZATION

Despite the apparent increased complexity of the generalized ResNet architecture, implementation is simpler than that of the orignal ResNet block and is immediately usable as a single linear operation in any existing architecture at no added cost in time or parameters. We implement the architecture with a modified initialization of a standard convolutional or fully connected layer that combines the identity shortcut with the desired linear transformation (convolution or matrix multiplication) followed by concatenating the residual and memory streams together (see Equation 2 for an example in the case of a fully connected layer).

$$\begin{bmatrix} \mathbf{r}_{l+1} \\ \mathbf{m}_{l+1} \end{bmatrix} = \sigma(\begin{bmatrix} W_{l,1} + I & W_{l,2} \\ W_{l,3} & W_{l,4} \end{bmatrix} \times \begin{bmatrix} \mathbf{r}_l \\ \mathbf{m}_l \end{bmatrix}) \tag{2}$$

2.2 RESNET IN RESNET (RIR)

The ResNet initialization described above applied to each convolution within an original ResNet unit leads us to a new building block we call ResNet in ResNet (RiR) (Figure 1d). In Figure 2, we summarize the relationship between standard CNN, CNN with ResNet Init, ResNet, and RiR architectures.

3 EXPERIMENTS

Initial results on CIFAR-10 and CIFAR-100 show the RiR architecture is remarkably effective, obtaining competitive results on CIFAR-10 with only standard augmentation by random crops and horizontal flips, and state-of-the-art results on CIFAR-100. Our results indicate that ResNet Init consistently improves performance over a standard CNN without shortcut connections and that the

	initialization		
with shortcut connections		N	Y
	N	CNN	ResNet Init
	Y	ResNet	RiR

Figure 2: Relationship between standard CNN, ResNet, ResNet Init, and Resnet in Resnet.

Table 1: Test set accuracy of baseline 32 layer CNN (He et al., 2015) on CIFAR-10.

Model	Accuracy (%)
ResNet (He et al., 2015)	92.49
CNN	89.03
ResNet	92.32
ResNet Init	89.62
RiR	**92.97**

Table 2: Comparison of our architecture with state-of-the-art architectures on CIFAR-10.

Model	Accuracy (%)
Highway Network	92.40
ResNet (32 layers)	92.49
ResNet (110 layers)	93.57
Large ALL-CNN	95.59
Fractional Max-Pooling	**96.53**
18-layer + wide CNN	93.64
18-layer + wide ResNet	93.95
18-layer + wide ResNet Init	94.28
18-layer + wide RiR	94.99

Table 3: Comparison of our architecture with state-of-the-art architectures on CIFAR-100.

Model	Accuracy (%)
Highway Network	67.76
ELU-Network	75.72
18-layer + wide CNN	75.17
18-layer + wide ResNet	76.58
18-layer + wide ResNet Init	75.99
18-layer + wide RiR	**77.10**

RiR architecture outperforms the standard ResNet (Table 1). All models used standard ReLU activations and batch normalization (Ioffe & Szegedy, 2015) and we report the best result found after a grid search on hyperparameters including learning rate, L2 penalty, initialization, and the type of shortcut connections (identity or projection) used when increasing dimension in the ResNet and RiR architectures. In Table 4, we try combinations of different numbers of blocks and layers in each block. We find that the RiR architecture performs well across a range of hyperparameters. In Table 5, we show the ResNet Init applied in existing architectures, such as ALL-CNN-C (Springenberg et al., 2014), yields improvement over standard initialization.

We also investigate the effect of ResNets on a shallower and wider 18-layer network, and find they perform extremely well (Tables 2 and 3). These results indicate benefits of residual architectures do not derive solely from training networks of increased depth.

To test the hypothesis that the ability to remove information improves residual architectures, we experiment with networks that incorporate forget gate layers similar to those in an LSTM (Hochreiter & Schmidhuber, 1997), in which the gate is computed by a single convolutional layer followed by a sigmoid. Preliminary results from this architecture in Table 6 are extremely promising and we are continuing to run experiments. Other experiments in progress include testing the effects of different sizes of residual and working memory streams used in the generalized residual network architectures and visualization of how information is blended across streams.

4 Conclusion

We present a generalized residual network architecture that improves ResNets with the ability to forget information and learn deeper residuals, can be simply implemented by a modified initialization scheme, and can be plugged into any existing architecture. We demonstrate improvement on computer vision baselines from applying our architecture to several models and achieve state-of-the-art results. Future work includes analyzing the performance of this initialization on other architectures, comparing models with heavier data augmentation and ensembling to get an accurate comparison toother state-of-the-art models, and further studying RiR, ResNet Init, and related residual models to determine the cause of their beneficial effects.

Table 4: Performance of different numbers of blocks and layers per block of RiR on CIFAR-10.

# Blocks	Layers/Block	Accuracy (%)
15	2	92.87
3	10	90.06
6	5	92.98
15	5	92.11
9	**3**	**93.23**

Table 5: Performance of ALL-CNN-C from Springenberg et al. (2014) with batch normalization (Ioffe & Szegedy, 2015). The original model without batch normalization had 90.92% accuracy.

Model	Accuracy (%)
Standard initialization	93.22
ResNet Init	93.42

Table 6: Performance of baseline 32 layer CNN (He et al., 2015) on CIFAR-10 with and without forget gate layers.

Model	Accuracy (%)
ResNet	92.32
RiR	92.97
ResNet + forget	**93.36**
RiR + forget	93.07

REFERENCES

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin A. Riedmiller. Striving for simplicity: The all convolutional net. *CoRR*, abs/1412.6806, 2014. URL `http://arxiv.org/abs/1412.6806`.

Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. In *Advances in Neural Information Processing Systems*, pp. 2368–2376, 2015.