Gating Mechanisms in CNNs Inspired by Memorisation and Generalisation

Image Team, GREYC, ENSICAEN, UNICAEN, CNRS

All Models Not Surpassing Baseline Performance

Prachi Garg
September, 2019

In this work, we explore the affect of several gating mechanisms on the Resnet CNN architecture with two primary objectives:

1. To devise a deep CNN architecture that will handle the in-distribution and out-of-distribution examples differentially by relegating certain deeper layers to be used more exclusively by the hard examples while the easy examples skip them.
2. To understand qualitatively how the network can be made to generalise/memorise differentially and whether this will result in a performance increase from the baselines.

RELATED WORK

The idea of routing information through a DNN on information highways using a gating mechanism for better optimisation and easier training was first introduced in Highway Networks (ref. 6). This was extensively studied by (ref. 2). The skip connection experiments in the paper (ref. 2) give us a good starting point to understand the kind of gating strategies that have been explored for residual units in the past, their capacity and affect on the overall network performance. It helps us validate behaviour of different gating mechanisms that we experiment with.

LAYER LEVEL GATING EXPERIMENTS

Gating at the layer level refers to using gated skip connections instead of identity skip connections at each residual unit in a Resnet. Hence, in these architectures, there is a gate for each residual unit.

We start by conducting preliminary experiments on CIFAR with Resnet 110 and Resnet 164 based on the paper Identity mappings in deep residual networks (ref. 2). The reproduced baseline models and the Pre-residual models on Cifar 10 and Cifar 100 are given in tables 1.1 and 1.2. The training schedule used is specified by training scheme 1 under implementation details. We first reproduce the exclusive gating and shortcut-only gating experiments on CIFAR. It is evident from the results that manipulating the identity skip connections in this manner leads to optimisation issues and doesn't enhance performance.

TRAINING WITH REGRESSION LOSS

We next train models with the same exclusive and shortcut-only gates but with an additional regularisation term computed as L2 norm of g(x). This added loss is averaged over the L2 norm of g(x) values from all residual units (54) in the network. The total loss is back propagated through the entire network during back propagation.

We use a gate loss term as a regularisation term to enable the model to generalise better and attain a parsimonious solution in accordance with the rule of Ockham's razor. Intuitively, this will encourage the model to use lesser number of layers in the prediction of majority of the samples as there is a penalisation on using the layers.

Model	Mean test accuracy	Standard deviation (accuracy)	Median	Paper's claim, mean	mean training error
Resnet 110	93.478	0.2644239021	93.59	93.39	7.35E-06
Preresnet 110	93.786	0.1221883792	93.78	93.63	5.98E-06
Resnet 164	94.494	0.1108151614	94.54	94.07	6.79E-06
Preresnet 164	94.946	0.1492648653	94.96	94.54	6.35E-06

Table 1.1 Cifar 10 baselines and Pre-residual models averaged over 5 runs

Model	Mean test accuracy	Standard deviation (accuracy)	Median	Paper's claim,mean	Mean training error
Resnet 110	71.816	0.1820164828	71.76	NA	0.000078272
Resnet 164	74.466	0.9917308103	74.63	74.84%	0.000060864
Preresnet 164	75.804	0.4873704956	75.78	75.67%	0.000042808

Table 1.2 Cifar 100 baselines and Pre-residual models averaged over 5 runs

Model	Test accuracy	Paper’s claim	Training error
Resnet 110, exclusive gating	92.87	91.3	0.4085
Resnet 110, exclusive gating, with l2norm loss, lambda=0.01	93.24	NA	0.3468
Resnet 110. Shortcut only gating	93.57	93.09	0.3723
Resnet 110, shortcut only gating, with l2norm loss, lambda=0.01	93.57	NA	0.3389

Table 1.3 Cifar 10 skip connections experiments, (single run)

Model	Test accuracy	Training error
Resnet 164, exclusive gating	40.19	713.4
Resnet 164, exclusive gating, with g_l2norm loss	53.64	0.863
Resnet 164. Shortcut only gating	73.52	3.274
Resnet 164, shortcut only gating, with g_l2norm loss	72.3	0.015

Table 1.4 Cifar 100 skip connections experiments, (single run)

$Total\ Loss = Classification\ Loss + \lambda/n(\sum_{1}^{n} g(x)^2)$, *where n is number of residual units in the Resnet base architecture*

The skip connections results can be found in tables 1.3 and 1.4. All performance results above are reported after training on entire train set (50k images). They have been trained using the same learning scheme as the baselines. It has been observed that performance degrades on using the L2 norm loss on this architecture.

LAYER LEVEL 2-WAY CROSS ENTROPY GATING

Here we use 2 values g1(x) and g2(x) instead of a single g(x) value to control the gate. g1(x) and g2(x) are obtained from the gating layers at each residual unit.

$$Gate\ Output = g1(x) * ResidualUnitOut + g2(x) * x$$

For each residual unit, we experiment with 2 types of gating layer implementations:

1. Use a 1x1 convolution with 2 filters as the gating layer. The output is a 3D tensor with depth 2. The 2 gate feature maps are reshaped and used as g1(x) and g2(x) in the gate after passing the flattened 2D tensor through a softmax. The tensors g1(x) and g2(x) are also used to calculate the cross entropy loss with the targets specified in a manner that favours using the identity skip connection over the output from that unit's layers (all targets are set to the column index corresponding to g2(x) in the concatenated [g1(x) ,g2(x)] tensor). This is assuming that when g2(x)=1, the layers in that unit are skipped and when g1(x)=1, the output from the unit's layers is passed forward.
2. Use a 3x3 convolution with 2 filters as gating layer. Rest of the implementation is same as implementation (1)

$$Total\ Loss = Classification\ Loss + \lambda / n \sum_1^n (Gate\ Cross\ Entropy\ Loss)$$

Trained on a 45k/5k train to validation split ratio. Uses training scheme 1, trained for 250 epochs, only initialisation from scratch has been explored for this architecture. Trained for 9 lambda values ranging from 1e-5 to 1e+3 (increasing by a factor of 10 each time). Implementation (1) performs best for lambda=1e-5; implementation (2) gives best performance at lambda=1e-2. But the models with best performing hyper parameters are also poorer than the baselines.

BLOCK LEVEL GATING EXPERIMENTS

In this category of experiments, we apply inter residual unit skip connections. The idea is to skip multiple residual units in contrast to skipping a convolutional layer inside the same residual unit which is the case in layer level gating. In our experiments, this type of skip connection gating mechanism has been applied over a block of consecutive residual units. We conducted experiments on 2 types of Resnets for cifar 10 and cifar 100 datasets (ref. 1):

I. Imagenet type Resnet 34 with [3,4,6,3] residual units in subsequent blocks
II. Cifar type Resnet 110 and Resnet 164 with [18,18, 18] residual units in subsequent blocks

EXPERIMENTS ON IMAGENET TYPE RESNETS

Baseline Resnet 34 model trained for 164 epochs gives test_accuracy = 93.04%, (paper claims 92.5% with same training schedule, but on Resnet 32, which has [5, 5, 5] residual units in subsequent blocks).

In these architectures, we applied the skip connection and gating mechanism only over block 3 consisting of 6 residual units. The skip connection is the output of the 1st residual unit of B3, named o1. We specify o2 to be the output the last residual unit of B3. The gate is a soft gate, the gating value g(x) controls the affect of o1 and o2 on the output of the gate which in turns forms the input of the last block (B4). The idea is that the hard samples can have higher influence on the 5 remaining residual units where they can be memorised as compared to the easy samples that will skip the layers in question.

MODEL 1.

To get the gating value g(x), we flatten the output of the 1st residual unit of B3 and pass it through a fully connected block (512 unit FC followed by a single unit FC) which outputs a scalar value g(x). This is used as the gate control value after passing it through a sigmoid function. The L2 norm of g(x) is used to compute a gate loss term which is added to the total loss. Total loss is back propagated through the entire network. The gating mechanism is same as the one employed for resnet 110 in Figure 1.

$$Gate\ Output = g(x) * o2 + (1 - g(x)) * o1$$
$$Total\ Loss = Classification\ Loss + \lambda \sum g(x)^2$$

We conducted experiments on cifar 10 for this architecture. We used 2 initialisation strategies; first, we initialise the layers randomly using kaiming He initialisation and refer to this as scratch initialisation; second, we initialise the models with the baselines trained on the same dataset. The results are given in tables 2.1 and 2.2. they perform slightly better than the baseline and a lambda value of 0.01 gives best performance.

lambda	Test Accuracy	Training loss
lambda=0.01	**93.36%**	**0.6168**
lambda=1e-5	93.28%	1.007
lambda=1e-4	93.16%	0.8833
lambda=1.0	92.90%	1.259
lambda=1e-3	92.83%	1.006
lambda=0.1	92.80%	1.244
lambda=10.0	92.45%	1.716
lambda=100.0	87.96%	8.266
lambda=1000.0	82.48%	24.26%

Table 2.1: Model 1, L2 norm of g, training from scratch

lambda	Test Accuracy	Training loss
lambda=0.01	**93.46%**	**0.8773**
lambda=1e-5	92.98%	0.7847
lambda=1e-4	92.96%	1.001
lambda=1.0	92.95%	1.258
lambda=1e-3	93.35%	0.7033
lambda=0.1	92.85%	1.166
lambda=10.0	92.50%	1.537
lambda=100.0	88.67%	2.487
lambda=1000.0	81.42%	24.81

Table 2.2: Model 1, L2 norm of g, training from baseline

MODEL 1.2

We also trained model 1 by using norm between o1 and output of gate as gate loss term instead of norm of g. The architecture remains same as model 1.

$$Gate\ Output = o2 * g(x) + o1 * (1 - g(x))$$
$$Total\ Loss = Classification\ Loss + \lambda \sum [o1 - (o2 * g(x) + o1 * (1 - g(x)))]^2$$

When trained with a constant lr=0.1, it gives 95.312% test accuracy.

MODEL 2

The gating mechanism is applied over block 3 but the gating value g(x) is obtained in a different fashion. We flatten the output of the 1st residual unit of B3 and pass it through a fully connected block (512 unit FC followed by a 10 unit FC). We use the output of the FC passed through a softmax function to compute an entropy metric. This entropy value is passed through a sigmoid function to get the final gating value. We also experiment with another version where the entropy is not passed through sigmoid. No additional gate loss is used.

$$Total\ Loss = Classification\ Loss$$

Table 2.3 shows the results of experiments conducted with different initialisation strategies and choice of usage of sigmoid after entropy calculation.

Model	Test Accuracy	Training loss
With sigmoid, scratch	93.06%	0.5628
With sigmoid, baseline init	**93.43%**	**0.5653**
Without sigmoid, scratch	93.23%	0.5236
Without sigmoid, baseline init	92.90%	1.352

Table 2.3 - Model 2: using entropy as g

Training schedule used for all gating experiments on Resnet 34 is specified by training scheme 1. given under the implementation details section.

EXPERIMENTS ON CIFAR TYPE RESNETS

We next performed experiments on Resnet 110 and Resnet 164 as these are the standard resnet baselines for cifar dataset and it gives us an established base to compare our gating mechanisms with.

Resnet 110 and Resnet 164 have 3 blocks, each with 18 residual units stacked together. In all these experiments, unless specified, the gate is applied only over the last block (B3). Downsampling using stride 2 convolutions is performed only at the 1st residual unit of each block. The output feature map dimensions of all subsequent residual units in a block remain same. In order to have the skip connection and block output with the same dimensions in the gate, we use the output of the 1st residual unit as the skip connection.

The gates have been designed in a manner that the in-distribution samples will use the skip connection and skip the entire block (except the 1st residual unit of the block) and the out-of-distribution samples will pass through the block in order to utilise more model parameters offered by the layers in the block under consideration. The idea of using an additional loss term comprising the gate control parameters (g(x)) as a regularisation term is that the outliers pay a price for using the extra layers.

MODEL 1.

This gating mechanism is exactly same as model 1 of Imagenet type resnet experiments. Using a FC block with a singular scalar output which is used as the gate control variable g(x) after passing it through sigmoid. L2 norm of g(x) is used to compute gate loss term. Refer to Figure 1. For the architecture.

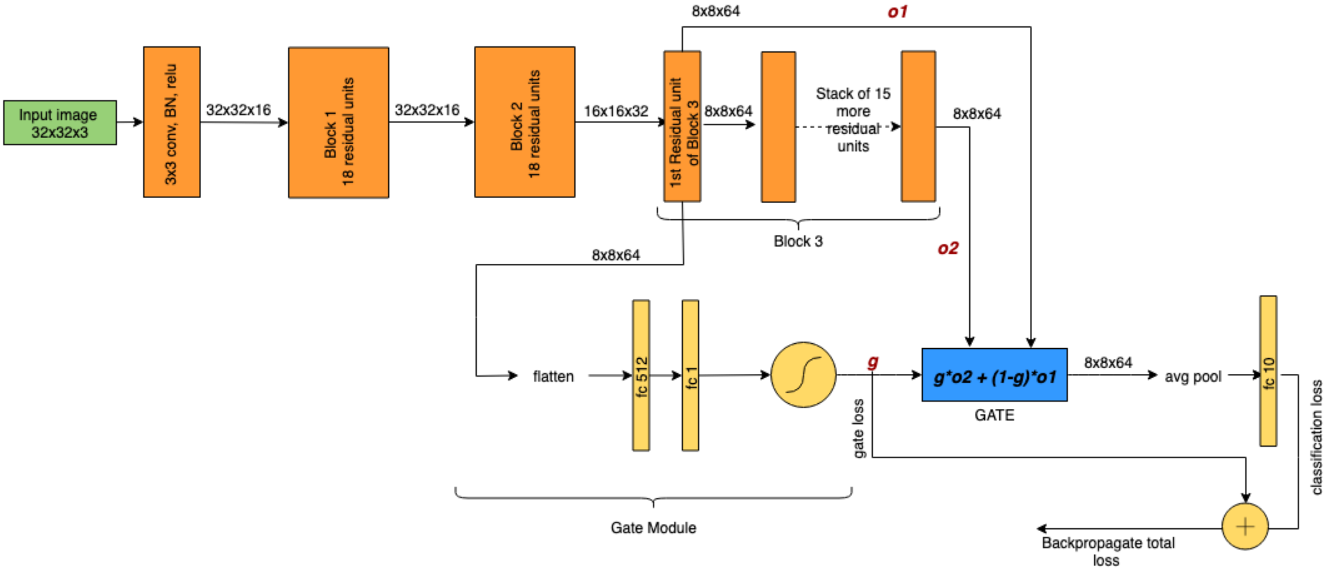

Figure 1. Network architecture for block level gating Model 1. This example applies the gate over B3 of Resnet 110 for cifar 10

$Gate\ Output = g(x) * o2 + (1 - g(x)) * o1$, *Where o1 is output of 1st residual unit of B3, o2 is output of B3*

$$Total\ Loss = Classification\ Loss + \lambda \sum g(x)^2$$

Here, we initialise all models with their respective baselines trained on the same dataset as the one we wish to train on after incorporating gating modifications. We train for a total of 250 epochs on the entire train set using training scheme 1. Experiments have been conducted with 9 lambda

lambda	Test Accuracy	Training loss
1E-02	92.55	1.80E-05
1E-05	92.77	9.276E-06
1E-04	**92.78**	**1.095E-05**
1E+00	92.22	2.231E-05
1E-03	92.76	1.317E-05
1E-01	92.49	2.31E-05
1E+01	92.64	2.74E-05
1E+02	92.19	2.528E-05
1E+03	91.81	2.682E-05

Table 3.1: Model 1: CIFAR 10, L2 norm of g

lambda	Test Accuracy	Training loss
1E-02	72.78	1.4922E-04
1E-05	72.51	6.361E-06
1E-04	**73.57**	**4.142E-04**
1.0E+00	73.33	3.903E-04
1E-03	73.16	2.178E-04
1E-01	73.26	3.166E-04
1.0E+01	72.86	4.132E-04
1.0E+02	72.18	4.329E-04
1.0E+03	73.19	7.2820

Table 3.2: Model 1: CIFAR 100, L2 norm of g

values ranging from 1e-5 to 1e+3. Tables 3.1 and 3.2 enlist the results for Cifar 10 and Cifar 100 respectively. These models perform poorer than the baselines which has a 93.478% test accuracy on cifar 10 and 74.466% test accuracy on cifar 100.

MODEL 2

This gating mechanism is same as model 2 of Imagenet type resnet experiments. Using a 10 way softmax as output of gate FC, we calculate entropy which is used as g(x) value. This entropy value is scaled about the origin and passed through a sigmoid function to get the final gating value. No regularisation term is used in this model. The architecture is given in figure 2.

$$Total\ Loss = Classification\ Loss$$

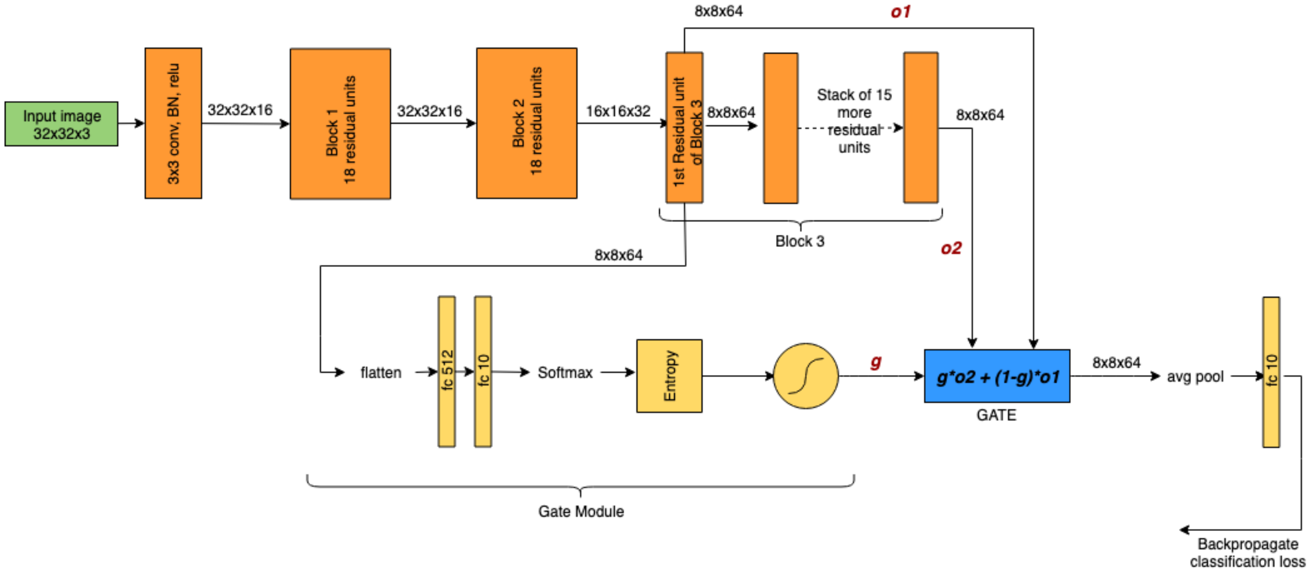

Figure 1. Network architecture for block level gating model 2. This example applies the gate over B3 of Resnet 110 for cifar 10

We only train models from their respective baselines and the results are shown in table 3.3.

Dataset	Test Accuracy	Training loss
Cifar 10	93.91	4.95E-06
Cifar 100	74.88	9.036E-05

Table 3.3: Model 2: using entropy as g, no extra loss term, training from baseline initialisation

MODEL 3

We next experiment with a gate where we use cross entropy loss to compute gate loss. The output of the gate FC is a 2 way softmax, one representing ‘skip’ or g1(x) and the other representing ‘do not skip’ or g2(x). While computing the gate cross entropy loss, we set the targets in such a manner that it favours the g1(x) or ‘skip’ output, meaning that we want majority of the samples to skip the block and only a few outliers to have an affect on the block layers.

$Gate\ Output = g1(x) * o1 + g2(x) * o2$, *Where o1 is the skip connection over the block 3 layers, o2 is the output of block 3*

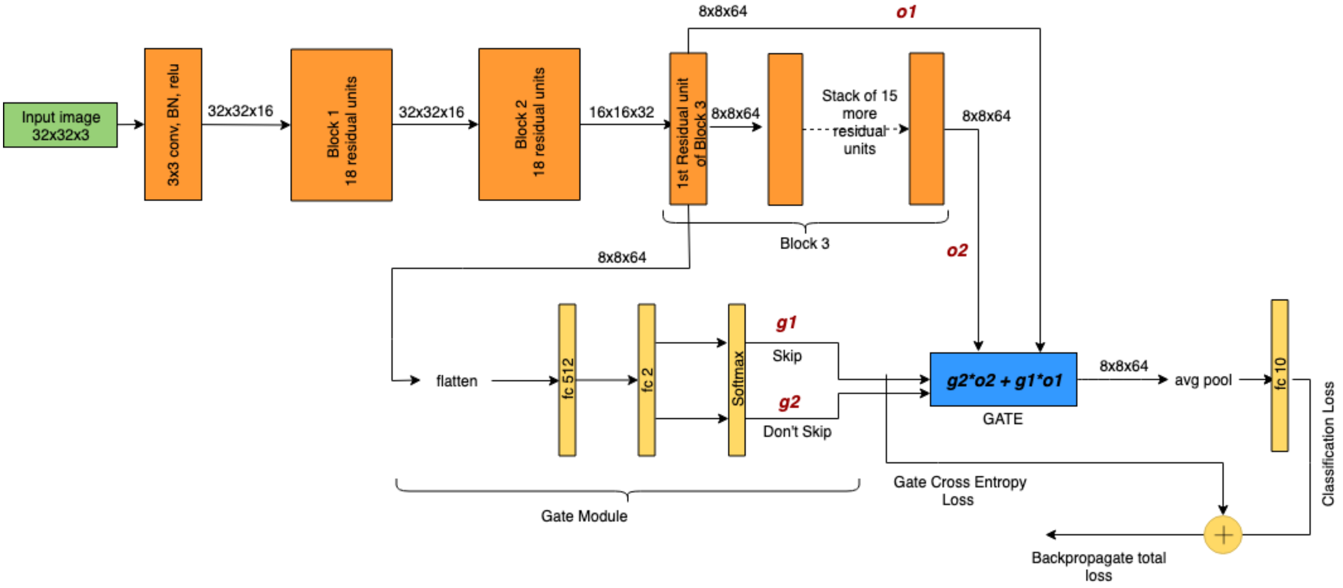

Figure 3. Network architecture for block level gating model 3. This example applies the gate over B3 of Resnet 110 for cifar 10

$$Total\ Loss = Classification\ Loss + \lambda \sum (Gate\ Cross\ Entropy\ Loss)$$

We experiment with the Pytorch inbuilt binary cross entropy loss and cross entropy loss. We use both initialisation from baseline and kaiming he initialisation from scratch for cifar 10 dataset. Cross entropy loss performs slightly better than the binary cross entropy loss. Training scheme 1 has been followed. The results are shown in tables 3.4, 3.5, 3.6, 3.7. However, the performance of these models fails to surpass their respective Resnet baselines.

lambda	Test Accuracy	Total Train Loss
1.00E-05	**92.66**	**9.13E-05**
1.00E-04	92.35	1.12E-05
1.00E-03	92.3	2.48E-05
1.00E-02	92.56	2.18E-05
1.00E-01	92.57	2.17E-05
1.00E+00	92.59	1.98E-05
1.00E+01	92.6	2.13E-05
1.00E+02	92.48	2.14E-05
1.00E+03	91.4	4.11E-05

Table 3.4: Model 3, cifar 10, scratch init, binary cross entropy gate loss

lambda	Test Accuracy	Total Train Loss
1.00E-05	92.32	9.71E-06
1.00E-04	**92.61**	**9.81E-06**
1.00E-03	92.6	1.82E-05
1.00E-02	92.44	2.24E-05
1.00E-01	92.49	2.30E-05
1.00E+00	92.34	2.20E-05
1.00E+01	92.41	2.14E-05
1.00E+02	92.53	2.34E-05
1.00E+03	91.74	3.15E-05

Table 3.5: Model 3, cifar 10, baseline init, binary cross entropy gate loss

lambda	Test Accuracy	Total Train Loss
1.00E-05	92.34	9.08E-06
1.00E-04	92.6	1.26E-05
1.00E-03	92.33	1.54E-05
1.00E-02	**92.68**	**2.43E-05**
1.00E-01	92.54	2.21E-05
1.00E+00	92.6	2.31E-05
1.00E+01	92.15	2.33E-05
1.00E+02	86.59	1.04E-03
1.00E+03	10	NAN

Table 3.6: Model 3, cifar 10, scratch init, cross entropy gate loss

lambda	Test Accuracy	Total Train Loss
1.00E-05	92.33	2.33E-05
1.00E-04	**92.97**	**1.15E-05**
1.00E-03	92.4	1.70E-05
1.00E-02	92.57	2.35E-05
1.00E-01	92.04	2.32E-05
1.00E+00	92.34	2.31E-05
1.00E+01	92.38	2.33E-05
1.00E+02	90.31	1.97E-04
1.00E+03	18.1	2.44E-02

Table 3.7: Model 3, cifar 10, baseline init, cross entropy gate loss

IMPLEMENTATION DETAILS

In accordance with (ref. 1), all Resnet 110 architectures use the basic block, with 6n+2 stacked weighted layers and all Resnet 164 architectures use the bottleneck block, with 9n+2 weighted layers.

Training Scheme 1:

Training schedule used to train these is similar to the one prescribed in the original papers (ref. 1,2). We use a weight decay of 0.0001 and momentum of 0.9, adopt the kaiming weight initialisation and BN with no dropout. These models are trained with a mini-batch size of 128 on a single GPU. For all CIFAR experiments on resnet depth greater than or equal to 110, we warm up the training by using a smaller learning rate of 0.01 at the beginning for 390 iterations and go back to 0.1 after that. We divide it by 10 at epochs 82 and 123, and train for a total of 250 epochs. Data augmentation for training: 4 pixels are padded on each side, and a 32×32 crop is randomly sampled from the padded image or its horizontal flip. For testing, we only normalise a single view of the image before using it.

Training error plateaus after initial 150 epochs when training the baselines which is not the case when training the gated models. Hence, all models have been trained upto 250 epochs (all gated models attain stable accuracy/training error when trained this way).

Note - Gating module or gating layers used throughout refers to the layer(s) that output the value used to control the gate, (commonly known as a ‘g(x)’). These layers output g(x) which in turn decides the fate of each sample as it passes through the gate

REFERENCES

1. Deep Residual Learning for Image Recognition, Kaiming he et. al.
2. Identity Mappings in Deep Residual Networks , Kaiming He et. al.
3. Understanding deep learning requires rethinking generalisation, Chiyuan Zhang et al.
4. A Closer Look at memorisation in Deep Networks, Devansh Arpit et al.
5. You look twice : GaterNet for dynamic filter selection in CNNs, Zhourong Chen et al
6. Highway networks, IDSIA