Highway Network Block with Gates Constraints for Training Very Deep Networks

Oyebade K. Oyedotun, Abd El Rahman Shabayek, Djamila Aouada, Björn Ottersten
Interdisciplinary Centre for Security, Reliability and Trust (SnT),
University of Luxembourg, L-1855 Luxembourg
{oyebade.oyedotun, abdelrahman.shabayek, djamila.aouada, bjorn.ottersten}@uni.lu

Abstract

In this paper, we propose to reformulate the learning of the highway network block to realize both early optimization and improved generalization of very deep networks while preserving the network depth. Gate constraints are duly employed to improve optimization, latent representations and parameterization usage in order to efficiently learn hierarchical feature transformations which are crucial for the success of any deep network. One of the earliest very deep models with over 30 layers that was successfully trained relied on highway network blocks. Although, highway blocks suffice for alleviating optimization problem via improved information flow, we show for the first time that further in training such highway blocks may result into learning mostly untransformed features and therefore a reduction in the effective depth of the model; this could negatively impact model generalization performance. Using the proposed approach, 15-layer and 20-layer models are successfully trained with one gate and a 32-layer model using three gates. This leads to a drastic reduction of model parameters as compared to the original highway network. Extensive experiments on CIFAR-10, CIFAR-100, Fashion-MNIST and USPS datasets are performed to validate the effectiveness of the proposed approach. Particularly, we outperform the original highway network and many state-of-the-art results. To the best our knowledge, on the Fashion-MNIST and USPS datasets, the achieved results are the best reported in literature.

1. Introduction

Many computer vision applications now rely on learning important features from data via deep neural networks [1] [2] as opposed to handcrafting such features. Over time, there has been a consistent challenge for better vision systems; for example, vision systems with lower error rates on image classification problems. Moreover, the complexity of the classification problems to be solved has consistently increased. For instance, about a decade ago, the MNIST handwritten digits dataset[1] was considered quite hard to learn. However, in the last 5 years, many deep learning based works [3] [4] have reported error rates in the range 0.5%-0.21%. Nevertheless, tackling more complex classification tasks has evolved a new direction for deep networks. This is in line with many theoretical works [5] [6] [7] that show the benefit of depth for learning complex and compositional target functions. Consequently, there are many works [8] [9] that have explored very deep models with up to 200 layers of feature abstractions. It has been identified that a major problem in the training of very deep networks[2] is the consistent dilution of features over the several layers of transformations as we go deeper[3] into the model [8] [10]. Therefore, many of the works [8] [9] [10] that have successfully trained very deep networks have relied on some ways of routing untransformed features from the earlier layers through the model. One of the earliest works referred to as a highway network [10] employed highway blocks with gating mechanism for routing untransformed lower layer features through the model to alleviate the problem of model optimization. In [8], residual network with shortcut connections of identity mappings for bridging the hidden layers of the model was proposed; this was shown to alleviate the difficulty of model optimization and also achieving impressive results.

The problem with very deep networks that rely on highway network blocks is that the network may find it difficult to learn feature transformations (new latent representations) which are important for generalization as training progresses. The biases of gating units of the highway network block are initialized to negative values at the start of training; this positions the gating units close to saturation. Although, this alleviates the difficultly of model optimization by mostly routing untransformed features via the high-

[1] http://yann.lecun.com/exdb/mnist/
[2] Deep models with over 10 layers
[3] We refer to layers closer to the output layer

way block, the gating units further go into saturation as training progresses. It is observed that this was reported in the same work [10]. Consequently, there is a reduction in the effective depth of models constructed using such highway blocks. In addition, employing a gate for a layer means roughly doubling the number of parameters for that particular layer. Hence, model overfitting is a concern.

In this paper, the problem of learning very deep networks using gating mechanisms as in the highway network blocks [10] is revisited. In particular, we address the aforementioned problems by learning the highway network block via the use of gate constraints such that new feature transformations can be effectively extracted. Also, we do not sacrifice the ease of model optimization. Our contributions are summarized as follows:

1. Effective use of model depth, since there is a natural learning of new latent representations for data as training progresses.
2. Drastic reduction in the size of model parameters and therefore potential to over-fit, since far lesser number of gates are required for learning.
3. Improve both model optimization and regularization, considering the required number of training epochs and test performance, respectively.

The proposed approach is validated on the CIFAR-10, CIFAR-100, Fashion-MNIST and USPS datasets. The results obtained show improvements over the original highway network and many state-of-the-art results. The rest of this paper is organized as follows. In Section 2, we discuss related works. Section 3 gives the background on highway networks along with the problem statement. In Section 4, we provide details of the proposed approach. Section 5 contains experimental results. The paper is concluded in Section 6.

2. Related work

2.1. Depth impact on model performance

Deep networks of few layers (i.e. 3-7 layers) have been successfully used for learning different tasks [11] [12] with interesting results. However, tackling more difficult tasks requires that we review the learning characteristics of existing models, with a view to extending their capacity for improved performance. For example, as at 2012, the state-of-the-art top-5 error rate reported on the ILSVRC2012 dataset was 15.3%, and was achieved with the AlexNet [13] with 60 million parameters. The AlexNet is a modest model with 5 convolution layers, 3 max pooling layers, 3 fully connected layers and 'interspersed' local constrast normalization and dropout layers. The winner of ILSVRC2013 object classification challenge relied on a modification of the AlexNet model, extending it up to 8 convolution layers; the model was referred to a ZFNet [14] with a top-5 error rate of 14.8%. The GoogleNet [15] with considerably extended model depth in comparison to the ZFNet won the ILSVRC2014 object classification challenge; the GoogleNet is a 22 layer model with different smaller convolutions that are called inception modules. The GoogleNet achieved an impressive top-5 error rate of 6.67%. It is interesting to note that the ILSVRC2014 object classification challenge runner up employed a 16 layer model that was referred to as VGG16 with a top-5 error rate of 7.3%. For the ILSVRC2015 object classification challenge, the best performance was achieved using the ResNet [16] with 152 layers; the model achieved an astounding top-5 error rate of 3.57%. A quick evaluation of how depth has impacted results on the ILSVRC object classification challenge is shown in Figure 1.

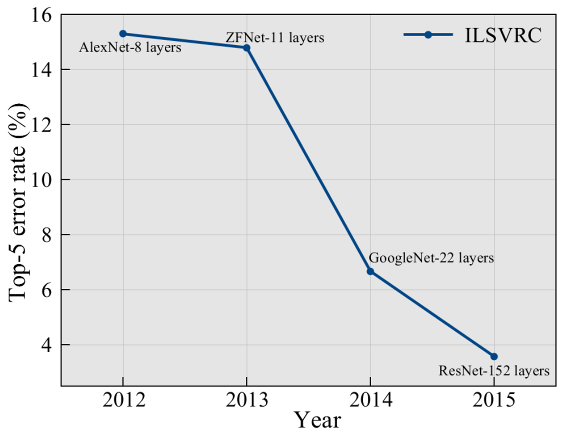

Figure 1. Depth impacts top-5 error rate (%) on the ILSVRC

Considering the discussion above on the evolution of the state-of-the-art results on the ILSVRC object classification challenge, the role of depth for deep networks in the learning of complex target functions becomes evident. We note that the same can be said for the impact of model depth on other popular and hard benchmarkinng datasets such as the CIFAR-10 and CIFAR-100 [8] [9] [17].

2.2. Very deep networks

Deep networks with few layers perform well on relatively simple tasks. For more complex tasks, empirical and theoretical evidences (as in Section 2.1) show that depth improves model performance; therefore, models with considerable depth (i.e. very deep networks) are required. However, different works [8] [9] [10] have reported training problems on models with more than 20 layers; specifically, fitting the training data (model optimization) is difficult. Generally, the problem is that learned features from the input layer get diluted over the many layers of feature transformation; that is, the features that reach the output layer are considerably unreflective of the input signal. Therefore, computed error gradients for updating model parameters have little impact in driving the model towards convergence. In [18], experiments showing that different model parameters initialization schemes and batch normalization do not resolve the difficulty of training very deep networks beyond some layers.

Consequently, the different works [8] [9] [10] that have successfully trained models with several layers have relied on different approaches for bypassing some feature transformations as information is routed from the input to the output layer. One of the pioneering works on very deep networks is the highway network [10] that successfully trained up to 50 layers. He et al. [8] proposed the residual network (ResNet) which employed shortcut connections of identity mappings for alleviating model optimization with depth; results for ResNets with over 100 layers were reported on the CIFAR-10 dataset. Also, Haung et al. [9] proposed to randomly drop a subset of the hidden layers of the ResNet by bypassing them using shortcut connections of identity mappings. The training scheme was reported to have improved training time and model regularization.

3. Background and problem statement

In this section, the background on the model that we build on is given; that is, the highway network [10]. Subsequently, the problem statement is presented.

3.1. Background: highway network

Training very deep networks has been a relatively long standing problem. Although, theory [6] [7] (and intuition) suggests that extending the levels of latent representations for deep models can lead to a more compact (or efficient) representation of highly varying target functions. The highway network is one of the first models with up to 50 layers to be successfully trained. The work [10] was reported to have taken inspiration from the Long Short Term Memory (LSTM) recurrent network [19] for constructing the highway network, as the model employs gating mechanisms for routing information from lower layers to higher layers.

The highway network block relies on gating mechanisms for controlling information flow via the model. Given that $H(x)^{l-1}$ is the information on the highway at layer $l-1$, the gating module outputs a signal $G^l(H(x)^{l-1})$ for controlling what information is routed to succeeding highway network block. $F^l(H(x)^{l-1})$ is the transformation learned at the hidden layer l for input $H(x)^{l-1}$; $H(x)^l$ is the final output of the highway network block at layer l and can be written as

$$\begin{aligned} H(x)^l = F^l(H(x)^{l-1})G^l(H(x)^{l-1}) + \\ H(x)^{l-1}(1 - G^l(H(x)^{l-1})). \end{aligned} \tag{1}$$

The form of information routing given in (1) was proposed and demonstrated to give promising results in [10]. In this formulation, the gate G^l at layer l can either allow or impede the flow of incoming signals $H(x)^{l-1}$ and $F^l(H(x)^{l-1})$, depending on the current state of the gating units; the gates can be either *opened* or *closed*. A diagram illustrating a block of a highway network is shown in Figure 2. In order to alleviate the difficultly of model optimization due to signal 'attenuation' with depth as discussed in Section 2.2, it follows that the gate units are initialized such that most of $H(x)^{l-1}$ (untransformed incoming features) are routed via the block early in training. That is, gates are opened early in training and therefore favour early model optimization. To achieve this, we have that:

- The gating units use the Log-Sigmoid function as the activation function so that units' outputs are rescaled to be within the range 0 to 1; where, states 0 and 1 denote closed and open gates, respectively. Note that the transformation path employs rectified linear units (ReLUs) or similar, as is typical for deep networks.

- At the start of training, the biases of the gating units are initialized to negative values such as -1 or -3 [10] so that $G^l(H(x)^{l-1})$ is very small (i.e. close to 0) and therefore most of $H(x)^{l-1}$ is routed via the highway block without transformation as the output $H(x)^l$.

The highway block shown in Figure 2 can be stacked to facilitate model optimization of very deep networks, since features routed from lower layers do not always undergo transformation.

3.2. Problem statement

In [8], the problem of fast model convergence and generalization was considered for proposing the residual network for which we learn a transformation of the form

$$H(x)^l = F^l(H(x)^{l-1}) + H(x)^{l-1}, \tag{2}$$

where the notations are the same as in (1).

One of the arguments presented in [8] for proposing the residual network is that both the transformed and untransformed features from lower layers are always routed via the model and therefore aid learning. In contrast, the highway network block transforms some features, but routes others through the model without transformation. This smooth transitioning from routing untransformed features to routing transformed features is a desirable property of the highway network. Particularly, [10] observed that gates deeper into the network are selective; that is, they perform some sort of feature filtering. We note that the residual network lacks this interesting attribute. Nevertheless, there are some concerns with learning the highway network as

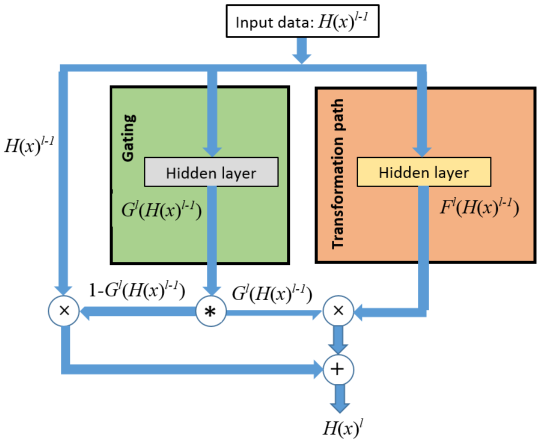

Figure 2. Highway network block [10]

discussed below.

First, we argue that the form of learning a highway block proposed in [10] and given in (1) is less natural. Initializing the gating units (that use Log-Sigmoid activation functions) to negative values at the start of training keeps the units considerably close to the saturation regime at the start of training. Although, this suffices for routing untransformed features from the lower layers to higher layers, this may not be the optimal way to employ the gates since it is well known that Log-Sigmoid units typically go into the lower end of the saturation 'spectrum' during training. Consequently, learning new transformations is largely impeded, since the gates mostly remain around the saturation regime (i.e. have very small output values) and therefore route only a small portion of the transformed features. Hence, $G^l(H(x)^{l-1})$ converges to zero as training progresses and subsequently we can write (1) as

$$H(x)^l \approx H(x)^{l-1} \quad \text{for} \quad i \gg 1 \mid 1 \leq i \leq t, \tag{3}$$

where i is the epoch index and t is the number of maximum epochs. This can impact model generalization since new features are not being effectively learned for data. Consequently, more training epochs can be required to drive the model to good convergence.
Furthermore, the original highway network [10] employed gates for every layer of feature transformation to alleviate the difficultly of model optimization. This considerably increases the parameters of the model, since a gate doubles the overall size of any layer where it is employed. As such, there is a huge risk of overfitting the training data. Moreover, employing many gates for highway networks could mean that the transformation path is less constrained to capture very important features in the training data, since there are enormous gate parameters that can hold data representations. Again, we note that this may impact the performance of the model at test time.

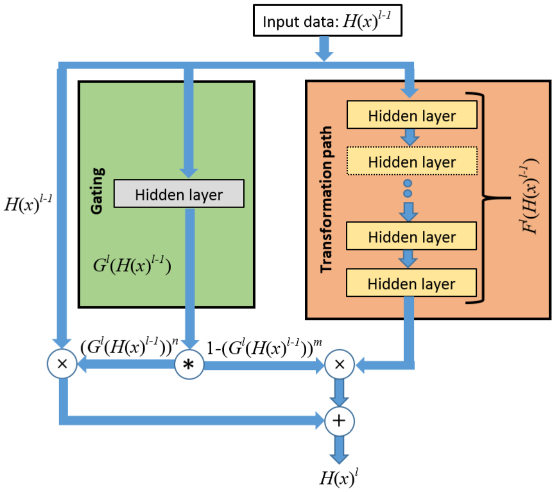

Figure 3. Proposed highway network block

4. Proposed approach

In this section, the details of the proposed approach are given, along with the different model components for addressing the problems of learning very deep networks that employ highway network blocks as mentioned in Section 3.2. In particular, the learning of the highway network block is reformulated to achieve the following:

(i) Favour optimization early in training; that is, the gates mostly route untransformed features at the start of training.

(ii) Focus on learning feature transformations later in training; that is, the gates mostly route transformed features.

(iii) Rely on far fewer gates for learning several layers of feature transformations, since gate transitioning is more effective for model optimization and generalization.

The interesting characteristic transitioning mentioned in (i) & (ii) allows the efficient use of model parameters and fewer gates for training as in (iii). As such, there is a lower risk of model over-fitting and therefore improved generalization. In order to achieve the aforementioned learning characteristics, we propose a new highway block of the form

$$\begin{aligned} H(x)^l = F^l(H(x)^{l-1})(1 - (G^l(H(x)^{l-1}))^m) + \\ H(x)^{l-1}(G^l(H(x)^{l-1}))^n, \end{aligned} \tag{4}$$

where m and n are model parameters which are introduced for remapping the gating function. The proposed model is shown in Figure 3. Note that the information on the highway is now gated by $(G^l(H(x)^{l-1}))^n$ as against

$1-(G^l(H(x)^{l-1}))$ that was proposed for the original highway network block [10]. In order to learn the highway block proposed in (4), and address the aforementioned challenges, we detail in what follows three necessary model components, namely, initialization, feature remapping, and gate constraining.

4.1. Model initialization

Gates initialization is very important to the training of highway networks as it determines the states of the gates for model optimization. At the start of training, it is usually desirable that the gates are open to favour early optimization. As with the earlier work [10], we rely on the biases of gating units for controlling the regime of operation of a highway block; that is, essentially routing untransformed or transformed features from the lower layers. However, our formulation requires that the biases of the gating units are initialized to relatively small values within the range 1 to 3. This places the gating units in the open state at the start of training. As it is conventional for Log-Sigmoid units, the biases of the gating units decrease as training progress and tend towards saturation. In order to increase the operation time of the proposed highway block for early optimization, a simple gating remapping discussed in Section 4.2 is performed.

4.2. Feature gating and re-mapping

In the proposed highway block, model parameters m and n as in (4) are used for constraining the gates to operate longer in the open state regime to favour optimization. The gating units use the Log-Sigmoid function as activation function, therefore gate units have outputs bounded as $0 < G^l(H(x)^{l-1}) < 1$. To ensure that model optimization prevails early in training, we constrain $n < 1 \leq m$; that is, gate closure is delayed as the gating units go into saturation. The remapping (or scaling) of gating outputs from $G^l(H(x)^{l-1})$ to $(G^l(H(x)^{l-1}))^n$ for the information highway path $H(x)^{l-1}$ for different values of n is shown in Figure 4. The arrows in Figure 4 show that going with the initialization method given in Section 4.1, the output of a gating unit and its remapped value decreases as training progresses. The values of m and n can be set during training by using a validation set. Note that m and n determine how much of transformed and untransformed features are routed via the highway block, respectively.

4.3. Gate constraint

The motivation for reversing the gating of the highway features and transformed features is that it allows us to easily put learning constraints on the gates. For the proposed highway block, the weights of the gating units are constrained to a max–norm of zero; that is, $\parallel \overrightarrow{w} \parallel < 0$, where w is the weight vector of an arbitrary gating unit. This considerably limits the representations that the gating units can learn and subsequently enforces the transformation path to learn important features (or representations). Consequently, the gating units essentially perform gating operation. Putting together the components of our proposed approach, it can be seen that much later in training, the gating units go into saturation (as it is typical for deep networks); that is, $G^l(H(x)^{l-1}) \longrightarrow 0$. Hence, we can write (4) as

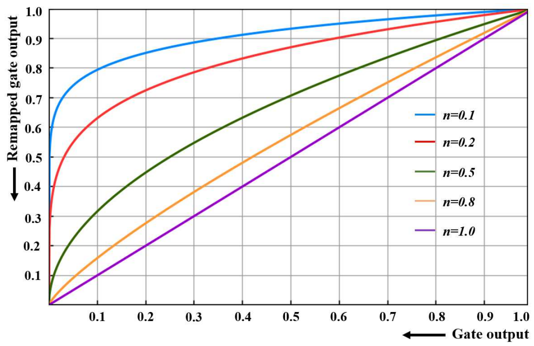

Figure 4. Gate remapping

$$H(x)^l \approx F^l(H(x)^{l-1}) \quad \text{for} \quad i \gg 1 \mid 1 \leq i \leq t, \qquad (5)$$

where notations remain the same as previously stated.
The important implications of the proposed highway block are as follows:

(i) Further in training, the highway block essentially learns feature transformations as in (5) which are critical for model generalization at test time. This contrasts with the original highway network block that mostly learns untransformed features and essentially remains in the optimization regime even much further in training.

(ii) The extreme scenario of our approach suggests that the proposed model starts out as a relatively 'shallow' model and mostly routing untransformed features. However, the model depth grows towards the effective depth as training progresses. At the end of training, we can 'roughly' recover a deep model without the effects of the gates as given in (5).

5. Experiments

In order to validate the learning capacity of the proposed highway block, very deep networks are constructed for performing extensive experiments using CIFAR-10[4] and CIFAR-100[4]. This allows the direct comparison of results of the very deep models that are learned using the proposed highway network blocks with the models learned using the original highway blocks [10]. Subsequently, we use other

[4] https://www.cs.toronto.edu/ kriz/cifar.html

32-layer highway architecture	
Output size	Input: 32×32×3
32×32	Conv1-64(3×3)
32×32	Conv2-32(1×1)
32×32	Conv3-32(3×3)
32×32	Conv4-64(1×1)
32×32	Conv5-48(1×1)
32×32	Conv6-48(3×3)
32×32	Highway__conv7-64(1×1)
32×32	Conv8-96(3×3)
16×16	MP__1(2×2); stride=2
16×16	Conv9-96(1×1)
16×16	Conv10-96(3×3)
16×16	Conv11-128(1×1)
16×16	Conv12-96(1×1)
16×16	Conv13-96(3×3)
16×16	Conv14-128(1×1)
16×16	Conv15-96(1×1)
16×16	Conv16-96(3×3)
16×16	Highway__conv17-128(1×1)
16×16	Conv18-192(3×3)
7×7	MP__2(3×3); stride=2
7×7	Conv19-192(1×1)
7×7	Conv20-192(3×3)
7×7	Conv21-256(1×1)
7×7	Conv22-192(1×1)
7×7	Conv23-192(3×3)
7×7	Conv24-256(1×1)
7×7	Conv25-192(1×1)
7×7	Conv26-192(3×3)
7×7	Highway__conv27-256(1×1)
7×7	Conv28-256(3×3)
2×2	MP__3(3×3); stride=3
1×1	2×2 global average pool
C-way softmax	

Table 1. Proposed 32-layer highway network architecture

standard datasets such as the Fashion-MNIST[5] and USPS[6] to further demonstrate the consistency of performance of the models learned using our approach; results comparison against state-of-the-art results are provided. The architecture of the 32 layer model used in this paper is given in table 1; where for convL-M(s×s), L denotes the convolution layer, M is the number of output feature maps and s is the size of convolution filter; Highway__convL-D(s×s) is a highway layer at layer L and with D output feature maps. For MP__N(k×k), N denotes the index of the max pooling layer and k is the size of the pooling window. For C-way softmax, C is the number of classes; for CIFAR-10 and CIFAR-100, C is 10 and 100, respectively. For the sake of compactness in reporting experimental results, we label the very deep networks learned using the proposed highway blocks with gate constraints as 'highway net.+GC' and the original highway network as 'highway net.'. The rest of this section contains the training settings of the constructed models for the different datasets used in this paper. In addition, experimental results, comparison with the state-of-the-art results and high level discussions are presented.

5.1. Training settings

We construct very deep networks using the proposed highway network block. Particularly, we experiment with 15, 20 and 32 layer networks. For the 15 and 20 layer networks, only a single gating layer is used for learning the whole model. For the 32 layer network, three gating layers are used for learning the whole model. For the constructed models, it is found that using the proposed highway block after every 8-13 layers suffices for alleviating the difficulty of model training. All the models are trained using mini-batch gradient descent optimization with a batch size of 128. For all our experiments, the biases of all gating units are initialized to 3. Also, using validation sets, it is found that setting m=1 and n=0.1 works well, giving very promising results on all the datasets; see tables 2 to 5. Dropout of various rates are applied only in the transformations layers for model regularization. In order to constrain the gating layers from learning important features but perform essentially feature gating, a max-norm of zero is applied on the weights of all gating layers. It is observed that though [10] did not report on the number of model parameters, our model uses 1 and 3 gates for learning the reported 20 and 32 layers models, respectively. Consequently, the highway networks that employ gates for every layer of feature transformation should have an overall larger number of model parameters as compared to the model proposed in this paper.

5.2. CIFAR-10

The CIFAR-10 dataset contains natural and colour images of size 32 × 32 pixels belonging to 10 different categories of objects for classification. The training set contains 50,000 samples, while the test set contains 10,000 samples. The results of our experiments without and with data augmentation are given in table 2 as C10 and C10+, respectively. For data augmentation, we follow the common protocol [8] [9] [10] of random horizontal flipping, translation by 4 pixels and reflection. Also, we did not whiten the images. The 20 and 32 layer models are trained for 200 and 250 epochs, respectively. Table 2 shows that the very deep networks learned using the proposed highway blocks outperform the very deep model learned using the original highway blocks. Figure 5 shows the train and test errors against training epochs for the 32-layer model with and without

[5] https://github.com/zalandoresearch/fashion-mnist

[6] http://www.cad.zju.edu.cn/home/dengcai/Data/MLData.html

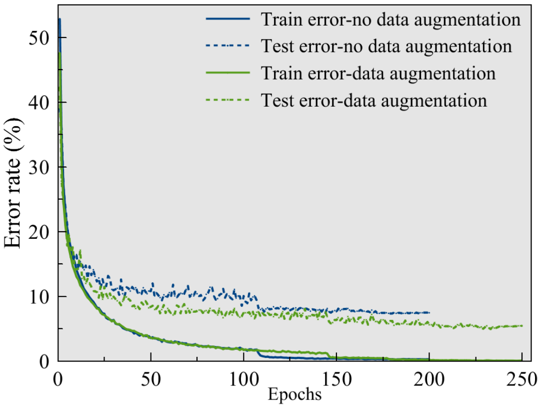

Figure 5. Train and test error on CIFAR-10

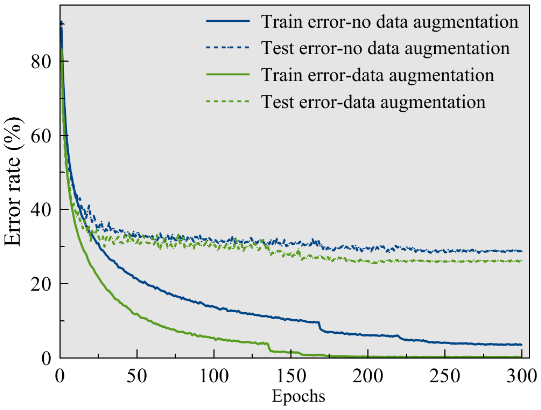

Figure 6. Train and test error on CIFAR-100

data augmentation. Our 32 layer model achieves error rates of 8.27% of 5.44% without and with data augmentation, respectively. We outperform the baseline model with a higher error rate of 7.72%. In addition, baseline model required 400 epochs for training, while the proposed models require much lesser number of epochs. Note that the result for the baseline model without data augmentation was not reported [10]. In addition, we outperform many state-of-the-art results. Since our aim is to demonstrate the effectiveness of the proposed highway blocks for learning very deep networks, we trained models of moderate depth and parameters. Nevertheless, the proposed models achieve very competitive results with extremely deep [8] [9] and wide [24] models. It is well known that the performance of deep models can be significantly improved by increasing model depth [8] [9] or width [24]. Consequently, the results given in table 2 can be improved by learning much deeper or wider models. However, this is at the cost of huge computational requirement. Although DenseNets [28] achieved lower error rates, we note that they requires much higher GPU (Graphics Processing Unit) memory [29].

5.3. CIFAR-100

The CIFAR-100 dataset contains natural and colour images of size 32 × 32 pixels belonging to 100 different categories of objects for classification. The training set contains 50,000 samples, while the test set contains 10,000 samples. The results of our experiments without and with data augmentation are given in table 3 as C100 and C100+, respectively. For data augmentation, the same protocol as in CIFAR-10 is followed. The 20 and 32 layer models are trained for 300 epochs. Table 3 shows that the very deep networks learned using the proposed highway blocks outperform the very deep model learned using the original highway blocks. Figure 6 shows the train and test errors against training epochs for the 32-layer model with and without data augmentation. Our 32 layer model achieves error rates of 29.26% and 25.26% without and with data

Models	Param.	Depth	C10	C10+
Network in network [20]	–	–	10.41	8.81
Recurrent CNN [21]	–	–	8.69	7.09
Deeply supervised [22]	–	–	9.69	7.97
All-CNN [23]	–	–	9.08	7.25
ResNet [9]	1.7M	110	13.63	6.41
Stochastic depth [9]	10.2M	110	11.66	5.23
Wide ResNet [24]	36.5M	28	–	4.17
Wide ResNet [24]	2.2M	40	–	5.33
Weighted ResNet [17]	19.1M	1192	–	5.10
FractalNet [25]	38.6M	21	7.33	4.60
DiracNet [26]	59M	34	–	4.54
DiracNet [26]	3.7M	34	–	5.60
Maxout [27]	>6M	–	11.68	9.38
DenseNet (k=24) [28]	27.2M	24	5.83	3.74
Baseline: Highway net. [10]	≫2.6M	32	–	7.72
Ours: highway net.+GC	1.7M	20	**9.88**	**7.08**
Ours: highway net.+GC	2.6M	32	**8.27**	**5.44**

Table 2. Error rate (%) on the CIFAR-10 dataset

Models	Param.	Depth	C100	C100+
Network in network [20]	–	–	35.68	–
Recurrent CNN [21]	–	–	31.75	–
Deeply supervised [22]	–	–	–	34.57
All-CNN [23]	–	–	–	33.71
ResNet [9]	1.7M	110	44.76	27.22
Stochastic depth [9]	10.2M	110	37.80	24.58
Wide ResNet [24]	36.5M	28	–	20.50
Wide ResNet [24]	2.2M	40	–	26.04
Fractional pooling [30]	–	–	–	27.60
FractalNet [25]	38.6M	21	29.05	23.73
DiracNet [26]	59M	34	–	20.96
DiracNet [26]	3.7M	34	–	26.72
Maxout [27]	>6M	–	38.57	–
DenseNet (k=24) [28]	27.2M	24	23.42	19.25
Baseline: Highway net. [10]	≫2.6M	32	–	32.39
Ours: highway net.+GC	1.7M	20	**29.81**	**26.15**
Ours: highway net.+GC	2.6M	32	**29.26**	**25.26**

Table 3. Error rate (%) on the CIFAR-100 dataset

Models	Param.	Depth	Fashion-MNIST
2C1P2F+Drouout [31]	3.27M	–	8.40
3C1P2F+Dropout [31]	7.14M	–	7.40
GoogleNet [31]	101M	22	6.30
AlexNet [31]	60M	8	10.10
SqueezeNet-200 [31]	0.5M	18	10.00
VGG16 [31]	26M	16	6.50
EvoCNN [31]	6.68M	–	5.47
Ours: highway net.+GC	1.7M	20	**5.26**

Table 4. Error rate (%) on the Fashion-MNIST dataset

augmentation, respectively. Again, the baseline model required 400 epochs for training, while the proposed models require 300 epochs.Note that the result for the baseline model without data augmentation was not reported [10]. We outperform the baseline model with a higher error rate of 32.39%. In addition, we outperform many state-of-the-art results. Also, note that the results that outperform ours have either significantly more depth [8] [9] or parameters [24]. We posit that the large improvement in performance as compared to the baseline model could be that since CIFAR-100 is a more difficult classification dataset, the proposed model allows the effective learning of feature transformations that are critical for model generalization. This contrasts with the baseline model that may find it difficult to learn new latent representations as training progresses.

5.4. Fashion-MNIST

The Fashion-MNIST dataset is a recent dataset for benchmarking learning algorithms. It was inspired from the fact that the popular MNIST handwritten digits dataset is now considered easy to learn considering recent reported results. The Fashion-MNIST dataset is similar to the MNIST, containing 50,000 and 10,000 training and testing samples, respectively. However, it contains images of fashion outfits belonging to 10 different classes which include T-shirt, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag and ankle boot. The images are grayscale and of size 28×28 pixels. We consider the Fashion-MNIST dataset to be simpler as compared to CIFAR-10 and CIFAR-100, therefore we train only a 20 layer model. Also, we perform no data augmentation. Table 4 shows that the model learned using the proposed highway blocks outperforms the state-of-the-art results including those with much more model parameters. Our 20 layer model achieves an error rate of 5.26% which to the best of our knowledge, this is the best result reported on the Fashion-MNIST dataset.

5.5. USPS

The USPS dataset contains handwritten digits from 0 to 9. Considering the state-of-the-art results [32] [18] [33] that have been reported so far on the USPS dataset, we note that this dataset is harder to learn as compared to the MNIST dataset. The USPS dataset contains 7,291 and 2,007 training and testing samples, respectively. The images are grayscale and of size 16×16 pixels. Again, we consider the USPS dataset to be simpler as compared to CIFAR-10 and CIFAR-100, therefore we train only a 15 layer model. Also, we perform no data augmentation. Table 5 shows the result of the model learned using the proposed highway blocks. Our 15 layer model achieves an error rate of 1.99% without data augmentation. We outperform several state-of-the-art results including human performance and those that employed data augmentation.

Models	Param.	Depth	USPS
Invariant vector supports [34]	–	–	3.00
LeNet [35]	–	5	4.20
NN + boosting [35]	–		2.60*
Manifold constraint transfer [36]	–	–	2.99
Evolutionary embedding [37]	–	–	3.90
Polynomial kernel SVM [38]	–	–	3.20
Tangent distance [32]	–	–	2.50*
Human performance [32]	–	–	2.50
Invariant scattering net.+PCA [33]	–	–	2.30
Residual network (ResNet) [18], [8]	–	54	3.34
Stochastic ResNet [18]	–	54	2.69
Ours: highway net.+GC	0.2M	15	**1.99**

*Result use data augmentation

Table 5. Error rate (%) on the USPS dataset

6. Conclusion

Very deep networks allow us to learn highly informative features from data. Training very deep networks is challenging due to model parameter optimization problems. One solution is to construct very deep networks by stacking several highway blocks to alleviate the difficulty of model optimization. Although the original highway block suffices for tackling the difficulty of model optimization, we argue that as training progresses, such highway blocks may mostly learn untransformed features and therefore negatively impact generalization capacity. In this paper, we reformulate the learning of highway blocks by employing gate constraints to improve model optimization and generalization. The proposed highway block for constructing very deep networks naturally lends itself to learning feature transformation as model training progresses. We perform experiments on CIFAR-10 and CIFAR-100 datasets to show that the very deep networks learned using the proposed highway block outperform very deep networks learned with the original highway blocks. Additionally, the results of the proposed model is competitive with respect to the state-of-the-art results for similar model depth and parameters. On the Fashion-MNIST dataset and USPS datasets, the proposed model achieves the best results.

Acknowledgments

This work was funded by the National Research Fund (FNR), Luxembourg, under the project reference R-AGR-0424-05-D/Bjorn Ottersten. This work was also supported by the National Research Fund (FNR), Luxembourg, under the project C15/IS/10415355/3D-ACT/Bjorn Ottersten, and FNR project IDform under the agreement C-PPP17/IS/11643091/IDform/Aouada.

References

[1] G. Papandreou, L.-C. Chen, K. P. Murphy, and A. L. Yuille, "Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1742–1750.

[2] Z. Liu, X. Li, P. Luo, C.-C. Loy, and X. Tang, "Semantic image segmentation via deep parsing network," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1377–1385.

[3] L. Hertel, E. Barth, T. Käster, and T. Martinetz, "Deep convolutional neural networks as generic feature extractors," in *Neural Networks (IJCNN), 2015 International Joint Conference on*. IEEE, 2015, pp. 1–4.

[4] J. Mairal, P. Koniusz, Z. Harchaoui, and C. Schmid, "Convolutional kernel networks," in *Advances in Neural Information Processing Systems*, 2014, pp. 2627–2635.

[5] O. Delalleau and Y. Bengio, "Shallow vs. deep sum-product networks," in *Advances in Neural Information Processing Systems*, 2011, pp. 666–674.

[6] H. Mhaskar, Q. Liao, and T. Poggio, "Learning functions: when is deep better than shallow," *arXiv preprint arXiv:1603.00988*, 2016.

[7] M. Bianchini and F. Scarselli, "On the complexity of neural network classifiers: A comparison between shallow and deep architectures," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 8, pp. 1553–1565, 2014.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[9] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *European Conference on Computer Vision*. Springer, 2016, pp. 646–661.

[10] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Advances in neural information processing systems*, 2015, pp. 2377–2385.

[11] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.

[12] A. de Brebisson and G. Montana, "Deep neural networks for anatomical brain segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2015, pp. 20–28.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[14] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*. Springer, 2014, pp. 818–833.

[15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[17] F. Shen, R. Gan, and G. Zeng, "Weighted residuals for very deep networks," in *Systems and Informatics (ICSAI), 2016 3rd International Conference on*. IEEE, 2016, pp. 936–941.

[18] O. K. Oyedotun, A. E. R. Shabayek, D. Aouada, and B. Ottersten, "Training very deep networks via residual learning with stochastic input shortcut connections," in *Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings*, vol. 10635. Springer, 2017, p. 23.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[20] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

[21] M. Liang and X. Hu, "Recurrent convolutional neural network for object recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3367–3375.

[22] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," in *Artificial Intelligence and Statistics*, 2015, pp. 562–570.

[23] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806*, 2014.

[24] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[25] G. Larsson, M. Maire, and G. Shakhnarovich, "Fractalnet: Ultra-deep neural networks without residuals," *arXiv preprint arXiv:1605.07648*, 2016.

[26] S. Zagoruyko and N. Komodakis, "Diracnets: Training very deep neural networks without skip-connections," *arXiv preprint arXiv:1706.00388*, 2017.

[27] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, “Maxout networks,” *arXiv preprint arXiv:1302.4389*, 2013.

[28] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, “Densely connected convolutional networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, vol. 1, no. 2, 2017, p. 3.

[29] Y. Zhu and S. Newsam, “Densenet for dense flow,” *arXiv preprint arXiv:1707.06316*, 2017.

[30] B. Graham, “Fractional max-pooling,” *arXiv preprint arXiv:1412.6071*, 2014.

[31] Y. Sun, B. Xue, and M. Zhang, “Evolving deep convolutional neural networks for image classification,” *arXiv preprint arXiv:1710.10741*, 2017.

[32] P. Simard, Y. LeCun, and J. S. Denker, “Efficient pattern recognition using a new transformation distance,” in *Advances in neural information processing systems*, 1993, pp. 50–58.

[33] J. Bruna and S. Mallat, “Invariant scattering convolution networks,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1872–1886, 2013.

[34] B. Schölkopf, P. Simard, A. J. Smola, and V. Vapnik, “Prior knowledge in support vector kernels,” in *Advances in neural information processing systems*, 1998, pp. 640–646.

[35] P. Simard, Y. LeCun, J. Denker, and B. Victorri, “Transformation invariance in pattern recognitiontangent distance and tangent propagation,” *Neural networks: tricks of the trade*, pp. 549–550, 1998.

[36] B. Zhang, A. Perina, V. Murino, and A. Del Bue, “Sparse representation classification with manifold constraints transfer,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 4557–4565.

[37] L. Liu, L. Shao, and X. Li, “Evolutionary compact embedding for large-scale image classification,” *Information Sciences*, vol. 316, pp. 567–581, 2015.

[38] S. Maji, A. C. Berg, and J. Malik, “Efficient classification for additive kernel svms,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 66–77, 2013.