TRAINING VERY DEEP NETWORKS

GROUP 15

E/13/087  E.W.L.B EGODAWELA
E/15/258  H.A.I.S PERERA
E/15/369  W.M.D UDANA

26/02/2021
Some Background Details

➢ Authors

◦ **Professor Jürgen Schmidhuber** (Father of modern AI) - University of Lugano, Switzerland
◦ Klaus Greff (Machine Learning PhD Student) - University of Lugano, Switzerland
◦ Rupesh Kumar Srivastava (Machine Learning PhD Student) - University of Lugano, Switzerland

➢ Location

◦ **The Swiss AI Lab** is a part of
  ◦ IDSIA Institution (Istituto Dalle Molle di Studi sull’Intelligenza Artificiale)
  ◦ USI University (Università della Svizzera Italiana)
  ◦ SUPSI University (Scuola Universitaria Professionale della Svizzera Italiana)

➢ Published in

◦ International Conference on Machine Learning (ICML) 2015 (+600 citations)

➢ Conference location and date

◦ Lille, France, 6 - 11 July 2015

Professor Jürgen Schmidhuber
Main Contributor to LSTM / Backpropagation
Addressing The Problem

➢ It is still an open problem why training becomes more difficult as depth of the neural network increases

➢ As number of hidden layers increases, number of inputs in each hidden layers increases

➢ This will make activation function less sensitive and become difficult to train neural network

➢ This problem is identified as **Vanishing Gradient Problem** in machine learning
Vanishing Gradient Problem

As more layers using certain activation functions are added to neural networks, the rate of the change of each activation function approaches zero.

Activation Function (Sigmoid) = \( \frac{1}{1 + e^{-z}} \)

Derivative = \( \frac{e^{-z}}{(1 + e^{-z})^2} \)
Introduction to Highway Networks

- It is a LSTM-inspired gating mechanism that information can flow across many layers without attenuation (more than 1000 layers)

- Highway Networks allow unimpeded information flow across many layers due to the Transform Gate (T) and Carry Gate (C)

\[
y = H(x, W_H) \cdot T(x, W_T) + x \cdot (1 - T(x, W_T))
\]
Introduction to Highway Networks

- It is based on RNN and LSTM neural network architectures

- Highway networks can be trained directly using Stochastic Gradient Descent (SGD) and does not stall for networks more than 1000 layer of network depth

- Gradient Descent is an iterative algorithm that start from a random point on a function and travels down its slope in steps until it reaches the lowest point of that function

- Stochastic Gradient Descent minimizes computations by picking data points randomly
RNN and LSTM Architectures

- Recurrent neural network architecture allows previous output to be used as input while having hidden states.
- RNN includes “Recurrent Cells” which can store information while processing new inputs.
- Long-Short term memory is the further developed RNN architecture.
- LSTM includes “Memory Cells” which can maintain information for long period of time.
Previous Work

- The top-5 (probability to be in top 5 predictions) image classification accuracy on the 1000-class ImageNet dataset has increased from 84% to 95% using deeper networks within just a few years due to the recent breakthrough.

- To deal with difficulties of training deep networks, some researchers have focused on developing better optimizers such as Hessian-free optimization (James Martens, Ilya Sutskever, 2012).

- Initialization strategies for activation functions, improvements in flow of information (shallow teacher network, Neural history compressor, Credit assignment problem) have been developed in recent years.
Contribution of This Research Paper

➢ To show that extremely deep highway networks can be trained directly using **Stochastic Gradient Descent (SGD)**

➢ Deep network with limited computational budget, such as training deeper student network in multiple stages, can be directly trained in a single stages
Highway Networks
Highway Networks

It is found that there are difficulties optimizing a very deep neural network. However, it’s still an open problem why it is difficult to optimize a deep network. Inspired by Long Short-Term Memory (LSTM), authors thereby make use of gating function to adaptively transform or bypass the signal so that the network can go deeper.
Plain network

Consider a plain feed forward neural network with \( L \) layers.

\( l^{th} \) layer applies a non linear transformation \( H \)

\[ y = H(x, W_H) \]

\( x \) is input, \( W_H \) is the weight, \( H \) is the transform function followed by an activation function and \( y \) is the output.

For \( i^{th} \) unit;

\[ y_i = H_i(x) \]

We compute the \( y_i \) and pass it to next layer.
In highway network, two non-linear transforms $T$ and $C$ are introduced:

$$y = H(x, W_H) \cdot T(x, W_T) + x \cdot C(x, W_C)$$

Where $T$ is the Transform Gate and $C$ is the Carry Gate.
For simplicity, $C = 1 - T$

$$y = H(x, W_H) \cdot T(x, W_T) + x \cdot (1 - T(x, W_T))$$

The dimensionality of $x$, $y$, $H(x, W_H)$ and $T(x, W_T)$ must be same.

Below conditions can be haven for particular $T$ values

$$y = x, \quad \text{if } T(x, W_T) = 0$$

$$y = H(x, W_H), \quad \text{if } T(x, W_T) = 1$$

When $T=0$, we pass the input as output directly which creates an information highway. That’s why it is called Highway Network !!!
When $T=1$, we use the non-linear activated transformed input as output

Highway network consists of multiple blocks such that $i$th block computes a block state $H_i(x)$ and transform gate output $T_i(x)$ and block output as;

$$y_i = H_i(x) \ast T_i(x) + x_i \ast (1 - T_i(x))$$
Constructing highway networks

The dimensionality of \( x, y, H(x, W_H) \) and \( T(x, W_T) \) must be same.

To change the size of intermediate representation:
- Can replace \( x \) with \( x \) obtain by sub sampling
- Use a plain layer

Convolutional highway layers utilize weight-sharing and local receptive field for both \( H \) and \( T \)

Same sized receptive fields for both and zero-padding are used to ensure that sizes will not change

\[
y = H(x, W_H) \cdot T(x, W_T) + x \cdot (I-T(x, W_T))
\]
Training deep highway networks

Formally, $T(x)$ is the sigmoid function

$$T(x) = \sigma (W_T x + b_T)$$

Highway Networks

$$\sigma(x) = \frac{1}{1+e^{-x}} \ , \ x \in \mathbb{R}$$
Highway Networks

\[ \sigma(x) = \frac{1}{1 + e^{-x}}, \quad x \in \mathbb{R} \]

The sigmoid function caps the output between 0 to 1. When the input has too small value, it becomes 0. When the input has too large value, it becomes 1. Therefore, by learning \( \mathbf{W}_T \) and \( \mathbf{b}_T \), the network can adaptively pass \( \mathbf{H}(\mathbf{x}) \) or just pass \( \mathbf{x} \) to next layer.

Here \( \mathbf{W}_T \) is the weight matrix and \( \mathbf{b}_T \) is the bias vector for the transform gates. This suggests a simple initialization scheme which is independent of the nature of \( \mathbf{H} \).

\( \mathbf{b}_T \) can be initialized with a negative value (e.g. -1, -3 etc.) such that the network is initially biased towards carry behavior.

This idea is inspired by LSTM as authors mentioned.
LSTM and Information Highways

LSTM

INFORMATION HIGHWAY

sigmoid  tanh  pointwise multiplication  pointwise addition  vector concatenation

Input  X  C  H  T  Output

Sigmoid  ReLU  Element-wise multiplication  Element-wise addition
Experiment
Optimization

- All networks were trained using SGD with momentum.

- An exponentially decaying learning rate was used in optimization:
  - Learning rate starts at a value $\lambda$ and decays according to a fixed schedule by factor $\Upsilon$.
  - $\lambda, \Upsilon$ and schedule were selected once based on validation performance on the CIFAR-10 and kept fixed for others.

- All convolutional highway networks utilize the ReLU activation function to compute $H$.

- Caffe and Brainstorm were used as frameworks.
Trained both plain and highway networks of varying depths on the MNIST digit classification dataset

All networks are thin:
- Each layer has 5 blocks for highway networks and 71 units for plain networks
- Yielding roughly identical number of parameters per layer

The first layer is a fully connected plain layer followed by 9, 19, 49, or 99 fully connected plain or highway layers. Finally, the network output is produced by a softmax layer.
Plain networks exhibit very good performance at 10 and 20 layers but it significantly degrades with depth increases.

Highway networks perform similar to the 10/20 layers networks at 50/100 layers.

It also consistently converged significantly faster than plain networks.
10-layer convolutional highway networks on MNIST are trained, using two architectures, each with 9 convolutional layers followed by a softmax output. The number of filter maps (width) was set to 16 and 32 for all the layers.

Compared with Maxout and DSN, Highway Networks obtained similar accuracy but with much fewer number of parameters.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10-layer (width 16)</td>
<td>10-layer (width 32)</td>
<td></td>
</tr>
<tr>
<td>No. of parameters</td>
<td>39 K</td>
<td>151 K</td>
<td>420 K</td>
</tr>
<tr>
<td>Test Accuracy (in %)</td>
<td>99.43 (99.4±0.03)</td>
<td>99.55 (99.54±0.02)</td>
<td>99.55</td>
</tr>
</tbody>
</table>
MNIST

Architecture of highway network for MNIST digits dataset

Input

9 convolutional highway hidden layers

Linear layer

Softmax

10 outputs
CIFAR10 & CIFAR100

- Fitnet cannot optimize the networks directly when the networks are deep. It needs two-stage training.
- By using gating function, Highway can optimize the deep networks directly. In particular, Highway B obtains highest accuracy with 19 layers.
- Though Highway C is inferior to Highway B, it stills can be optimized directly due to the existence of gating function.

<table>
<thead>
<tr>
<th>Network</th>
<th>No. of Layers</th>
<th>No. of Parameters</th>
<th>Accuracy (in %)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fitnet Results (reported by Romero et. al.[25])</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Teacher</td>
<td>5</td>
<td>~9M</td>
<td>90.18</td>
</tr>
<tr>
<td>Fitnet A</td>
<td>11</td>
<td>~250K</td>
<td>89.01</td>
</tr>
<tr>
<td>Fitnet B</td>
<td>19</td>
<td>~2.5M</td>
<td>91.61</td>
</tr>
<tr>
<td>Highway networks</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Highway A (Fitnet A)</td>
<td>11</td>
<td>~236K</td>
<td>89.18</td>
</tr>
<tr>
<td>Highway B (Fitnet B)</td>
<td>19</td>
<td>~2.3M</td>
<td>92.46 (92.28±0.16)</td>
</tr>
<tr>
<td>Highway C</td>
<td>32</td>
<td>~1.25M</td>
<td>91.20</td>
</tr>
</tbody>
</table>
Here, the fully connected layer used in the networks in the previous experiment is replaced with a convolutional layer with a receptive field of size one and a global average pooling layer.

Highway Network can obtain comparable performance on CIFAR-10 and highest accuracy on CIFAR-100
Comparison to Fitnets

- A maxout layer is simply a layer where the activation function is the max of the inputs.

- Maxout networks can cope much better with increased depth than those with traditional activation functions.

- Training on CIFAR-10 through plain backpropogation was only possible for maxout networks with a depth up to 5 layers when the number of parameters was limited to 250K.

- It is possible to obtain high performance on the CIFAR-10 and CIFAR-100 datasets by utilizing very large networks and extensive data augmentation.
The first hidden layer is a plain layer which changes the dimensionality of the representation to 50.

Each of the 49 highway layers (y-axis) consists of 50 blocks.

Visualization of best 50 hidden-layer highway networks trained on MNIST (top row) and CIFAR-100 (bottom row)

which were initialized to -2 and -4 respectively. In the second column the mean output of the transform gate over all training examples is depicted.

The third and fourth columns show the output of the transform gates and the block outputs for a single random training sample. Best viewed in color.
- For a single random sample for each transform gate respectively.
- Block outputs for the same single sample are displayed in the last column.
- The transform gate biases of the two networks were initialized to -2 and -4 respectively.

- The last column displays the block outputs and visualizes the concept of “information highways”. Most of the outputs stay constant over many layers forming a pattern of stripes.
- Most of the change in outputs happens in the early layers.
Routing of Information

- One possible advantage of the highway architecture over hard-wired shortcut connections is that the network can learn to dynamically adjust the routing of the information based on the current input.

- This behavior manifest itself in trained networks or do they just learn a static routing that applies to all inputs similarly.

- Learning to route information through neural networks with the help of competitive interactions.

- Very deep highway networks, on the other hand, can directly be trained with simple gradient descent methods due to their specific architecture.
Lesioned training set performance (y-axis) of the best 50-layer highway networks on MNIST (left) and CIFAR-100 (right).

As a function of the lesioned layer (x-axis). Evaluated on the full training set while forcefully closing all the transform gates of a single layer at a time.

The non-lesioned performance is indicated as a dashed line at the bottom.
A possible objection is that many layers might remain unused if the transform gates stay closed.

This experiments show that this possibility does not affect networks adversely—deep and narrow highway networks can match/exceed the accuracy of wide and shallow maxout networks.