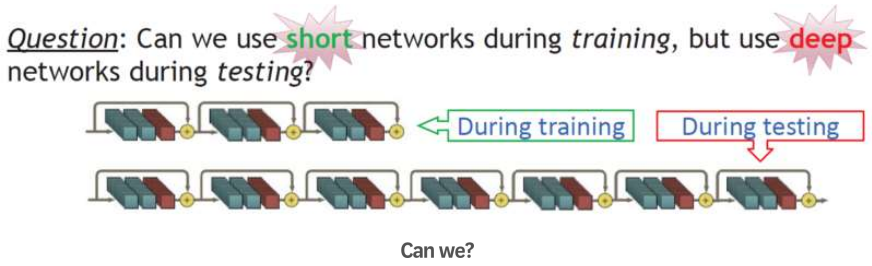
Review: Stochastic Depth (Image Classification)

SH Tsang Follow
Nov 27, 2018 · 5 min read

In this story, **Stochastic Depth** is shortly reviewed. Stochastic Depth, a training procedure to train short networks and use deep networks at test time. This is motivated by the dilemma:

> ***Using Deep Model****: We can get better prediction accuracy but it is difficult to train due to gradient vanishing problem.*

> ***Using shallower Model****: We can train the network more easily but the prediction accuracy sometimes is not good enough.*



By using Stochastic Depth, the network is shorten during training, i.e. a subset of layers is randomly dropped and bypass them with the identity function. And a full network is used during testing/inference. By this mean:

- **Training time is reduced substantially**
- **Test error is improved significantly as well**

This is a paper in **2016 ECCV** with about **400 citations** while I was writing story. (SH Tsang @ Medium)

. . .

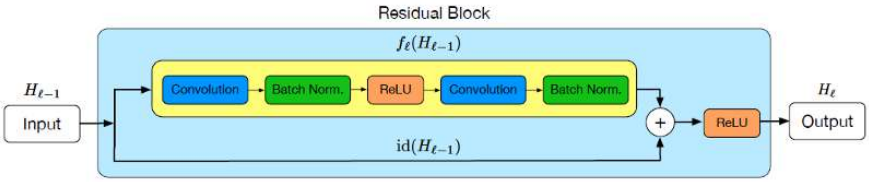
What Are Covered

1. **Brief Revision of Original ResNet**
2. **ResNet with Stochastic Depth**
3. **Some Results and Analysis**

. . .

1. Brief Revision of Original ResNet

$$H_\ell = \mathrm{ReLU}(f_\ell(H_{\ell-1}) + \mathrm{id}(H_{\ell-1}))$$



One l-th Residual Block (ResBlock) in Original ResNet

In **Original ResNet**, suppose *H_l-1* is the input of the above ResBlock, *H_l-1* will go through two paths.

- **Upper Path *f_l*(*H_l-1*)**: **Conv > BN > ReLU > Conv > BN**
- **Lower Path id(*H_l-1*)**: **Identity path** without modification of the input

Then the outputs of these two paths are **added together, then ReLU, and become *H_l*.**

By using the identity path, i.e. the skip connection or shortcut connection, we can keep the input signal, and attempt to avoid the gradient vanishing problem. Finally, we can obtain a very deep model.

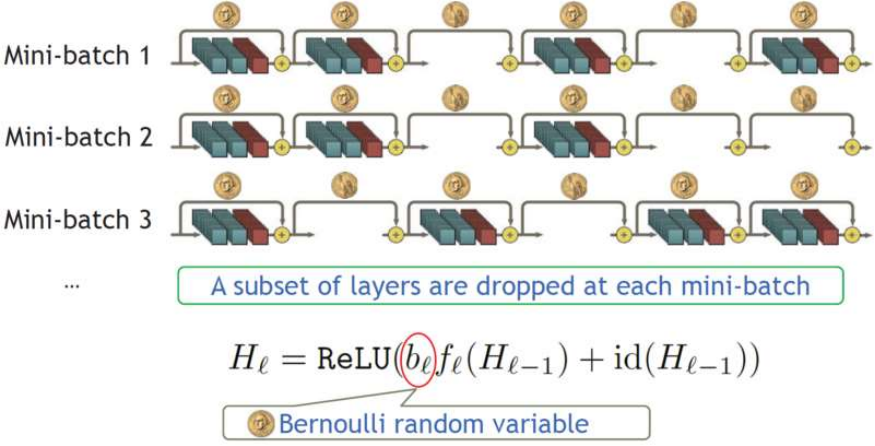
However, **training time** of such kind of deep models is **long**.

Also, there might be **overfitting problem**.

. . .

2. ResNet with Stochastic Depth

2.1. Training



Some ResBlocks are Dropped Randomly Based on Bernoulli Random Variable

$$b_\ell \sim \mathtt{Bernoulli}(p_\ell) \quad \text{with} \quad p_\ell = (1 - \frac{\ell}{L}) \times 1 + \frac{\ell}{L} \times p_L$$

Linear Decay Rule

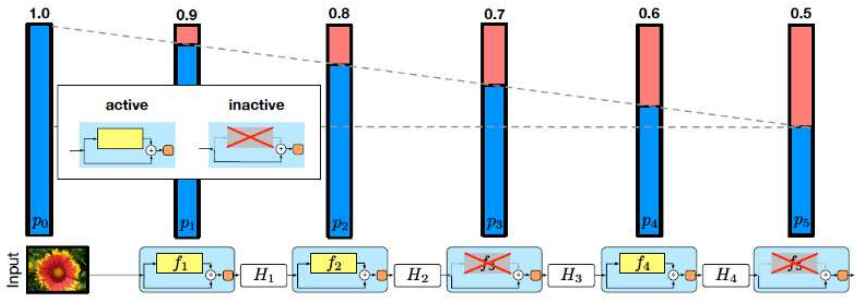
By using Stochastic Depth, during training, **for each mini-batch, each ResBlock would have a "survival" probability *p_l*.**

It is kept if survived. Otherwise it is skipped as shown above.

Networks trained with Stochastic Depth can be interpreted as an implicit ensemble of networks of different depths.

To decide *p_l*,

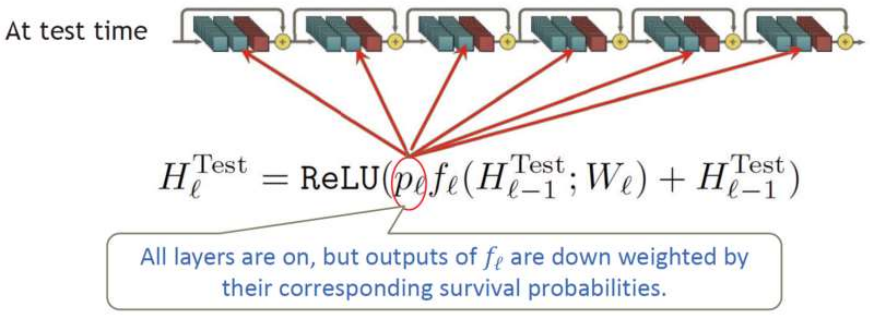
- One is to have **constant value** of *p_l* along the whole model.
- One is to have **linear decay rule** for *p_l* along the whole model.



If linear decay rule is used, earlier (deeper) layers have larger chance to be survived (skipped) as exampled above.

During training, the expected network depth is shorter than the whole network depth. If the whole network depth $L=110$, $p_L=0.5$, the expected network depth $E(L')=40$. **Therefore, training time is much shorter.**

2.2. Testing

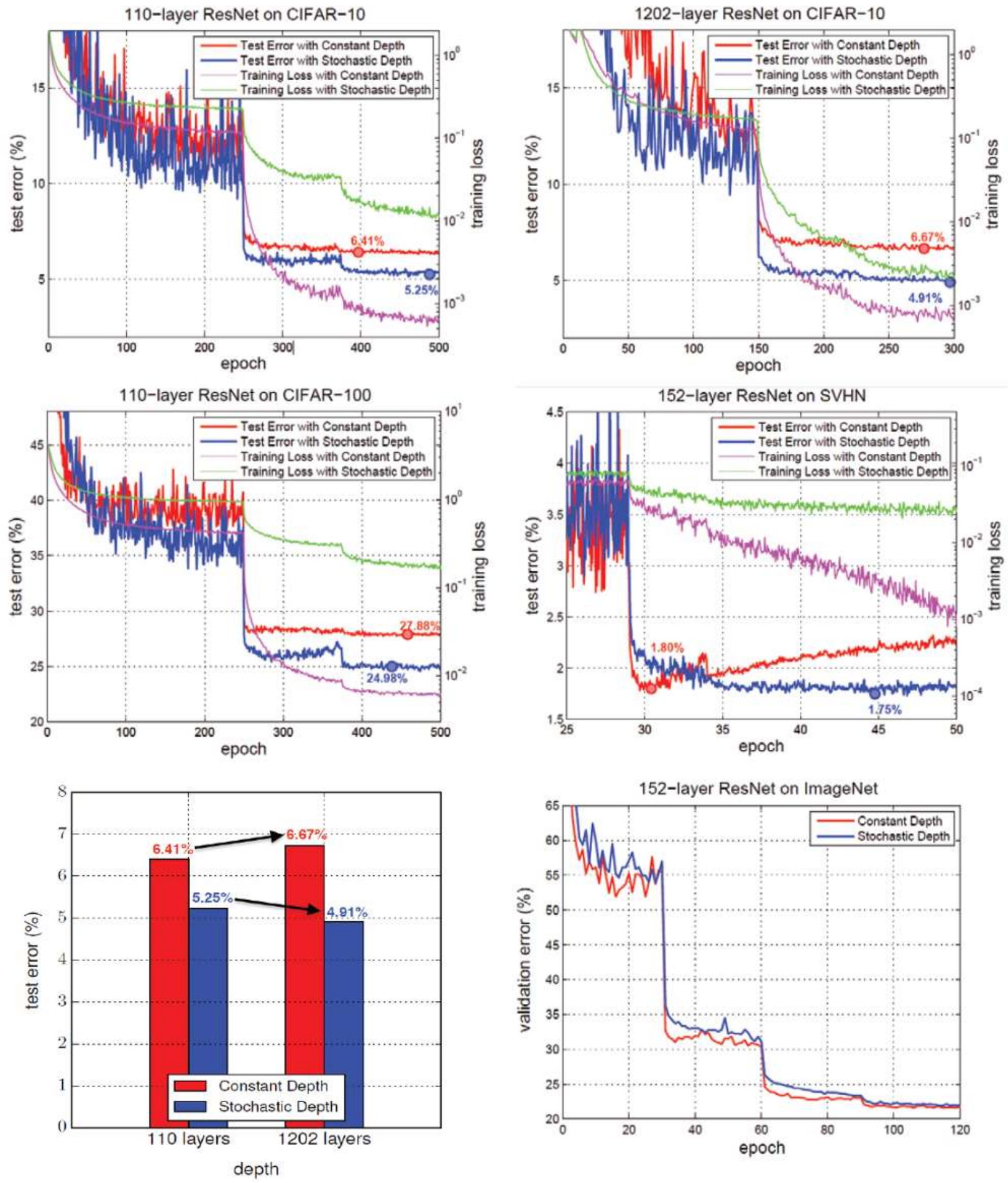


Weighted by p_l during testing

Stochastic depth during testing requires small modifications to the network. Since, during training, functions *f_l* are only active for a fraction *p_l* of all updates, and the corresponding weights of the next layer are calibrated for this survival probability. We therefore need to **re-calibrate the outputs of any given function *f_l* by the expected number of times it participates in training, *p_l*.**

. . .

3. Some Results and Analysis



CIFAR10, CIFAR100, ImageNet Results

3.1. CIFAR-10, CIFAR-100, SHVN

For example at top left of the figure:

- Training loss of constant depth < Training loss of Stochastic Depth
- Test loss of constant depth (6.41%) > Test loss of Stochastic Depth (5.25%)
- That means Stochastic Depth has reduced overfitting.

Similar trend for CIFAR-100 and SHVN using either 110 or 1202-Layer model.

	CIFAR10+	CIFAR100+	SVHN
Constant Depth	20h 42m	20h 51m	33h 43m
Stochastic Depth	15h 7m	15h 20m	25h 33m

~25% faster

Training Time

- Training time is much shorter as well.

3.2. ImageNet

For the ImageNet at the bottom right of the figure:

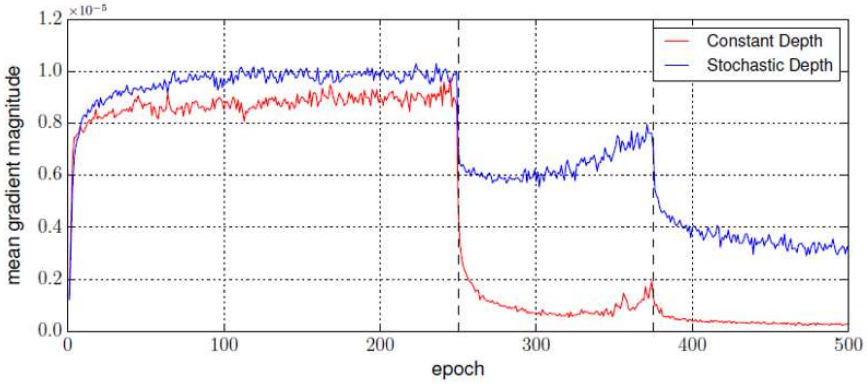
- Though the **validation error is very close** (23.06% for constant depth and 23.38% for Stochastic Depth), **training time is 25% shorter.**
- **If the training time is equal, this reaches a final error of 21.98%.**

Below is the detailed results for each dataset:

	CIFAR10+	CIFAR100+	SVHN	ImageNet
Maxout [21]	9.38	-	2.47	-
DropConnect [20]	9.32	-	1.94	-
Net in Net [24]	8.81	-	2.35	-
Deeply Supervised [13]	7.97	-	1.92	33.70
Frac. Pool [25]	-	27.62	-	-
All-CNN [6]	7.25	-	-	41.20
Learning Activation [26]	7.51	30.83	-	-
R-CNN [27]	7.09	-	1.77	-
Scalable BO [28]	6.37	27.40	1.77	-
Highway Network [29]	7.60	32.24	-	-
Gen. Pool [30]	6.05	-	1.69	28.02
ResNet with constant depth	6.41	27.76	1.80	21.78
ResNet with stochastic depth	5.25	24.98	1.75	21.98

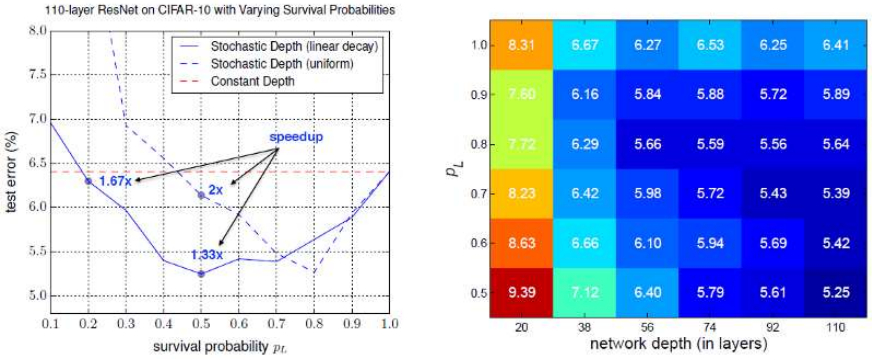
Detailed Results for Each Dataset

3.3. Analyses



Mean Gradient Magnitude

By looking at Mean Gradient Magnitude for each epoch, **Stochastic Depth has consistently larger weights than constant depth.** That means, **gradient vanishing problem is less severe in Stochastic Depth.**



Test Error vs Survival Probability (Left), and Test Error Heatmap with P_L vs Network Depth (Right)

For the Test Error vs Survival Probability at the left:

- Both assignment rules (linear decay and uniform assignment) yield better results than constant depth.
- Linear decay rule outperforms the uniform rule consistently.
- Linear decay rule obtains competitive results when *p_L* ranges from 0.4 to 0.8.
- With ***p_L*=0.2**, Stochastic Depth with linear decay still performs well, while giving a **40% reduction in training time**.

For the heatmap at the right:

- **Deeper networks with *p_L*=0.5 is better.**
- A deep enough model is necessary for stochastic depth to significantly outperform the baseline.

• • •

Other than dropout, we can also **drop some modules by using Stochastic Depth to decrease the network depth to reduce overfitting.**

. . .

References

[2016 ECCV] [Stochastic Depth]
Deep Networks with Stochastic Depth

My Related Reviews

[LeNet] [AlexNet] [ZFNet] [VGGNet] [SPPNet] [PReLU-Net] [GoogLeNet / Inception-v1] [BN-Inception / Inception-v2] [Inception-v3] [Inception-v4] [Xception] [MobileNetV1] [ResNet] [Pre-Activation ResNet] [DenseNet]