**Summary of the Paper**

**Deep Residual Learning for Image Recognition**

**Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun**

**Microsoft Research**

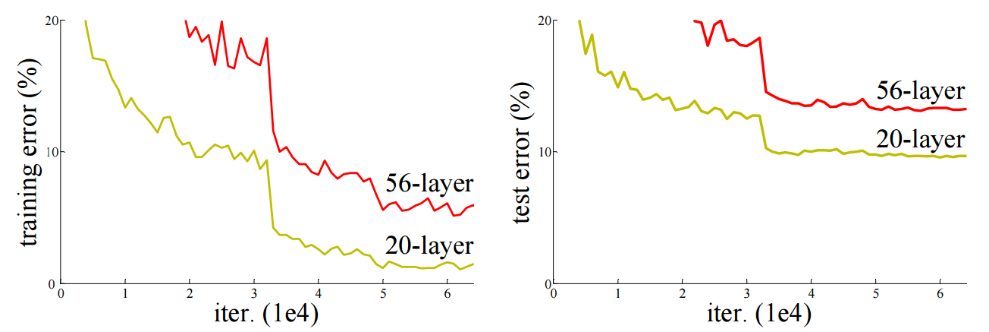
## Content Overview

We start with a short summary of the chapters in the paper. The following is a short introduction to deep convolutional networks, a mention of the related work used for the paper, an overview about residual learning, the used architecture and implementation and performance experiments on ImageNet 2012 as well as CIFAR-10.

### Introduction

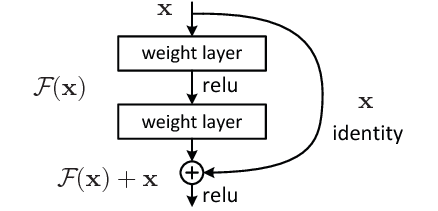
Each layer in a neural network detects a specific detail of the input image, depending on the filter kernel. Creating a deep neural network out of multiple layers stacked together allows the network to understand the correlation between those small details, leading to an abstraction of low-, mid- and high-level features. The paper refers to several sources, stating that a higher network depth is crucial for a better performance on the task of image recognition.

Surprisingly, the accuracy of deeper networks suffers from degradation. Not only on the testing data (which might be explained by overfitting), but also on the training data. The following figure demonstrates this behavior for the CIFAR-10 dataset:



Since it is possible to construct a solution for deeper networks, by copying the layers from a shallow network and only adding identity mappings to it, one might assume that the performance should be the same overall. The empirically proven deviation in performance suggests, that optimization of different networks is not similarly easy and that current optimization strategies are unable to find a comparably good solution in feasible time.

To compensate this the paper introduces a *deep residual learning* framework with a mapping F(x) + x. This formulation is achieved by a shortcut connection, that skips two convolutions, performs an identity mapping and is added to the output of the convolutions. As this simple identity shortcut does not add any parameters, the computational complexity is maintained, while leading to much better results. The building block for the deep residual learning network looks as follows:



### Related Work

### Deep Residual Learning

### Experiments

#### ImageNet 2012

#### CIFAR-10

## Comprehensibility

## Ease of Implementation

As a part of our assignment we implemented ResNet20 and ResNet32 from this paper and compared our results to the one in this paper.

### Implementation

At first implementing the stated ResNets was more complicated than we assumed. Especially because ResNet20 and ResNet32 are used for the CIFAR-10 Image-Set which uses a slightly different approach, then the residual nets described the main implementation part of the paper. It took a bit to figure out, which block-types were used, and which components were obsolete in the new residual nets. But once the first residual net was finished, it was to quite easy to add a different residual net.

A different problem was the runtime. In the paper 64 \*10^4 Iterations were performed. Even though we had a fast System available this was out of our reach, since the computation time, especially for deeper networks increased to multiple hours. Hence, we decided to compare the results of the first 30\*10^4 Iterations, which are shown in #TODO(Add Picture-reference). This however maid the exact use of the learning-rate-scheduler impossible, since it was used at 32\*10^4 and 48\*10^4 iterations. We decided to try a smaller approach and reduced the learning rate by factor of 0.1 at 16\*10^4 and 24\*10^4 iterations. Assuming we would get a better result short-term. Test showed this #TODO (ergebnis eintragen.)

### Success of Implementation

Overall the implementation of the two residual networks was very successful. Our results were almost identical with the ones from the paper. Additional we were able to show that increasing layers in residual networks also increase accuracy, while the opposite occurs in plain convolutional networks, where more layers lead to a decrease in accuracy.

One point we could not achieve was the overall amount of iterations because the computation took too long. We also stopped with two working neural networks, since additional layers also meant additional computation time.

## Comparision to Pytorch Implementation