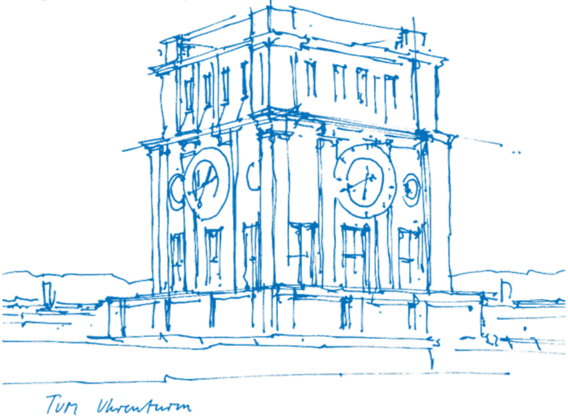
Deep Learning and Software

Image classification of CIFAR-10 with Deep Learning using TensorFlow

Lucas Schnack

Chair of Scientific Computing in Computer Science
Department of Informatics
Technical University of Munich

July 5th, 2022



Outline

TensorFlow [1]

- Open-source machine learning framework developed by the Google Brain team
- Good interaction with other Python libraries used for data science
- Uses tensors as its basic data type
- Data-flow oriented programming
- Offers a multitude of simple and complex implementations of various operations on these tensors
- Capable of automatic differentiation with gradient tapes

Keras [2]

- High-level interface for *TensorFlow*, official part of *TensorFlow* since 2019
- Enables modular, object-oriented development of neural networks
- Contains many implementations of commonly used layer types and statistical metrics (loss functions, etc.)
- Keras `Models` consist of `Layers`, which are connected to each other
- In the simple case of a feedforward neural network, these layers are just stacked in a linear order

```
model = Sequential()
model.add(Dense(10, input_shape=(10,)))
model.add(Dense(1))
```

Listing 1 A very simple fully-connected neural network

Outline

Convolutional Neural Networks
Convolutional layer

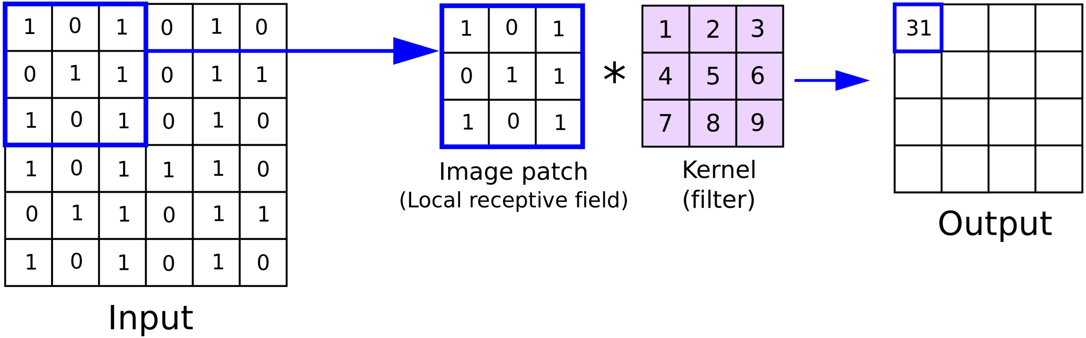


Figure 1 A convolutional layer. Reynolds, Anh H. 2019 [3]

Convolutional Neural Networks

Other layers

- Max Pooling: Similar to a convolutional layer, but just takes the maximum value instead of performing matrix operations with a kernel
  Reduces the size of the output
- Batch Normalization: Normalizes the input of a layer (re-scaling and re-centering)
  Helps to stabilize the training process, but is not yet fully-understood
- Simple convolutional neural networks usually consist of combinations of these three different layer types

Image Augmentation
Simple techniques

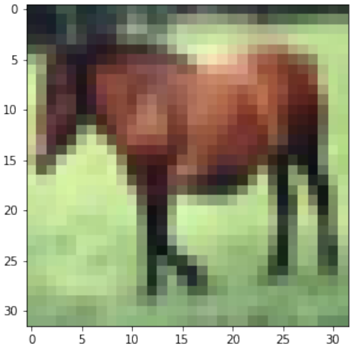


Figure 2 Original image

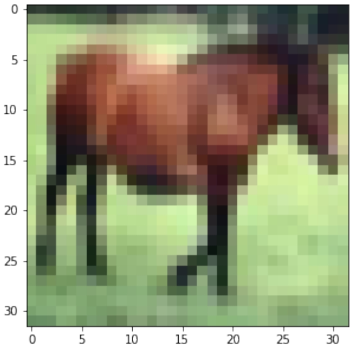


Figure 3 Flipped image

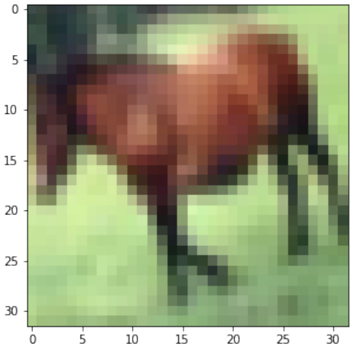


Figure 4 Rotated image

Image Augmentation
Implementation using Keras

```
# Create the augmentation layers
flip_layer = RandomFlip("horizontal")
rotate_layer = RandomRotation(0.15)

# Apply them to some image data
flipped = flip_layer(x)
rotated = rotate_layer(x)
```

Listing 2 Application of random flips and rotations to image data

CIFAR-10 [4]

- A collection of 60000 $32x32$ RGB color images
- 10 different image classes, e.g. airplanes, trucks, horses
- widely used for testing and comparing machine learning models

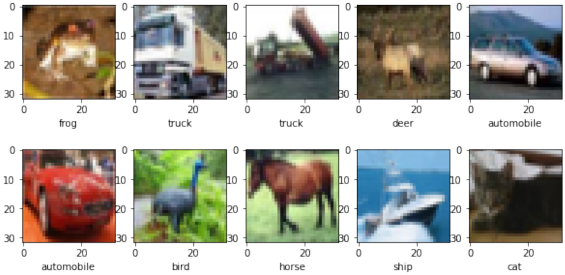


Figure 5 A sample from the CIFAR-10 dataset

Outline

Model implementation

- Implementation as a Keras model by subclassing `keras.Model`
- Usage of Keras layers for a simple feedforward neural network
- Network Architecture:
  1. Input layer
  2. Rescaling + Augmentation layer
  3. 4 x [Convolutional layer, Batch normalization, Convolutional layer, Batch normalization, Max Pooling layer]
  4. Flatten layer
  5. Dense layer
  6. Output layer

Training

A simple training loop consists of the following steps:

1. Fetch a batch of data (images and labels)
2. Generate predictions for the data using the model
3. Calculate the loss for the generated predictions
4. Compute the gradient of the loss function with respect to the trainable variables of the model
5. Use the gradient to update the trainable variables of the model using an optimizer
6. Repeat

A training loop like this is actually a one-liner in *Keras*:

```
model.fit(images, labels)
```

Listing 3 The Keras fit()-function

Outline

Variation of the learning rate

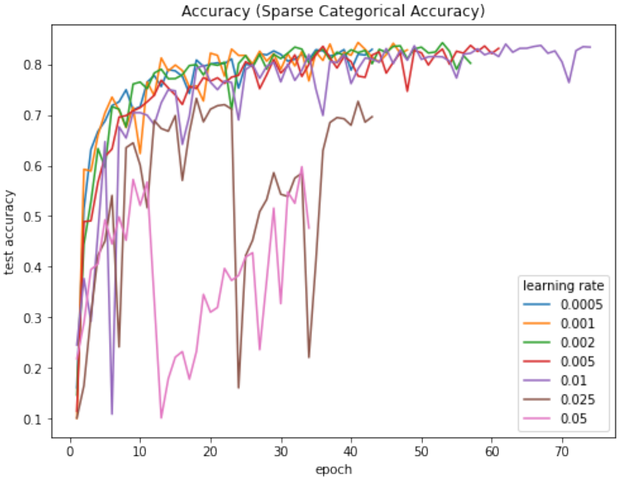
- One of the most important hyperparameters
- Controls the relative magnitude of the change in weights during the optimization process
- Impacts the convergence of the training process
- Trying different learning rates can be helpful to select an optimal one

Variation of the learning rate

learning rate	epochs until early-stopping	best test accuracy
0.0005	43	0.83
0.001	48	**0.8436**
0.002	57	0.8431
0.005	61	0.8378
0.01	74	0.8404
0.025	43	0.7331
0.05	34	0.5978

Table 1 Accuracy on the test data set for different learning rates

Variation of the learning rate



State-of-the-art

- Human accuracy is around 0.939 [5]
- Modern models achieve accuracies higher than 0.99 [6]
- Huge architectures with hundreds of millions of trainable parameters and hundreds of layers (compared to $1.3M$ parameters in our model)
- Vast resources and complicated techniques are required to train such models

Outline

Summary

- Modern machine learning frameworks like *TensorFlow* offer highly-efficient and easy-to-use ways to implement deep neural networks
- *Keras* can be used as a high-level interface for *TensorFlow*, customizing functionalities where necessary
- Convolutional neural networks are/were highly popular for deep learning with image data
- Varying the hyperparameters of the model and the training process can vastly impact its performance
- Huge and complex model architectures are able to surpass human performance at an increasing number of tasks

References I

- M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/
- F. Chollet *et al.*, "Keras," https://keras.io, 2015.
- A. H. Reynolds, "Convolutional neural networks (cnns)," 2019, accessed 07/05/2022. [Online]. Available: https://anhreynolds.com/blogs/cnn.html
- A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

References II

- T. Ho-Phuoc, “Cifar10 to compare visual recognition performance between deep neural networks and humans,” 11 2018.
- H. Touvron, M. Cord, A. Sablayrolles, G. Synnaeve, and H. Jégou, “Going deeper with image transformers,” 3 2021.