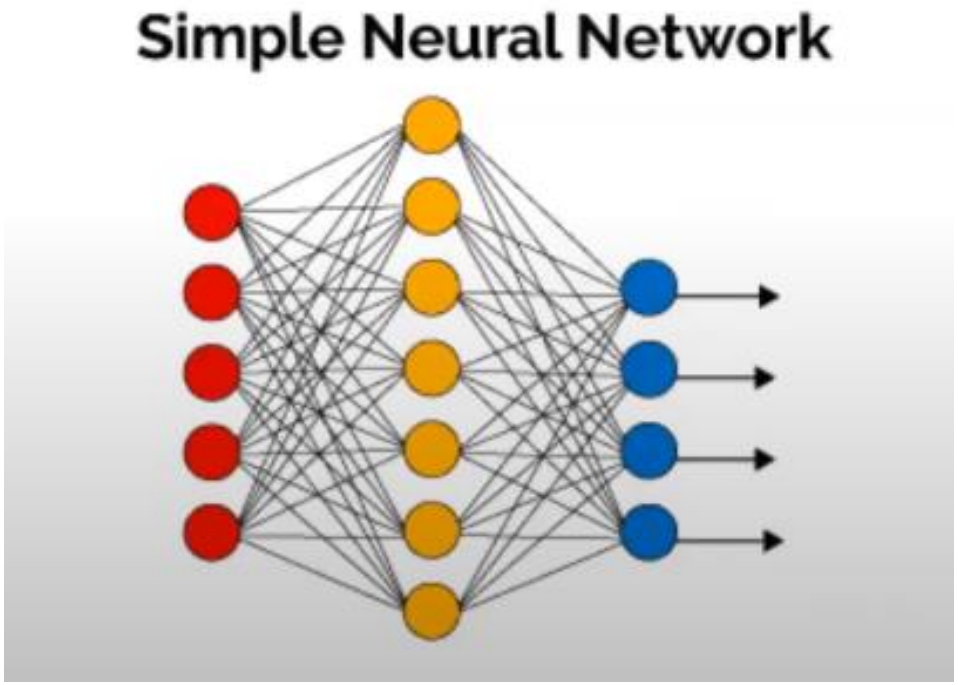
technische universität
dortmund

Recurrent Neural Networks

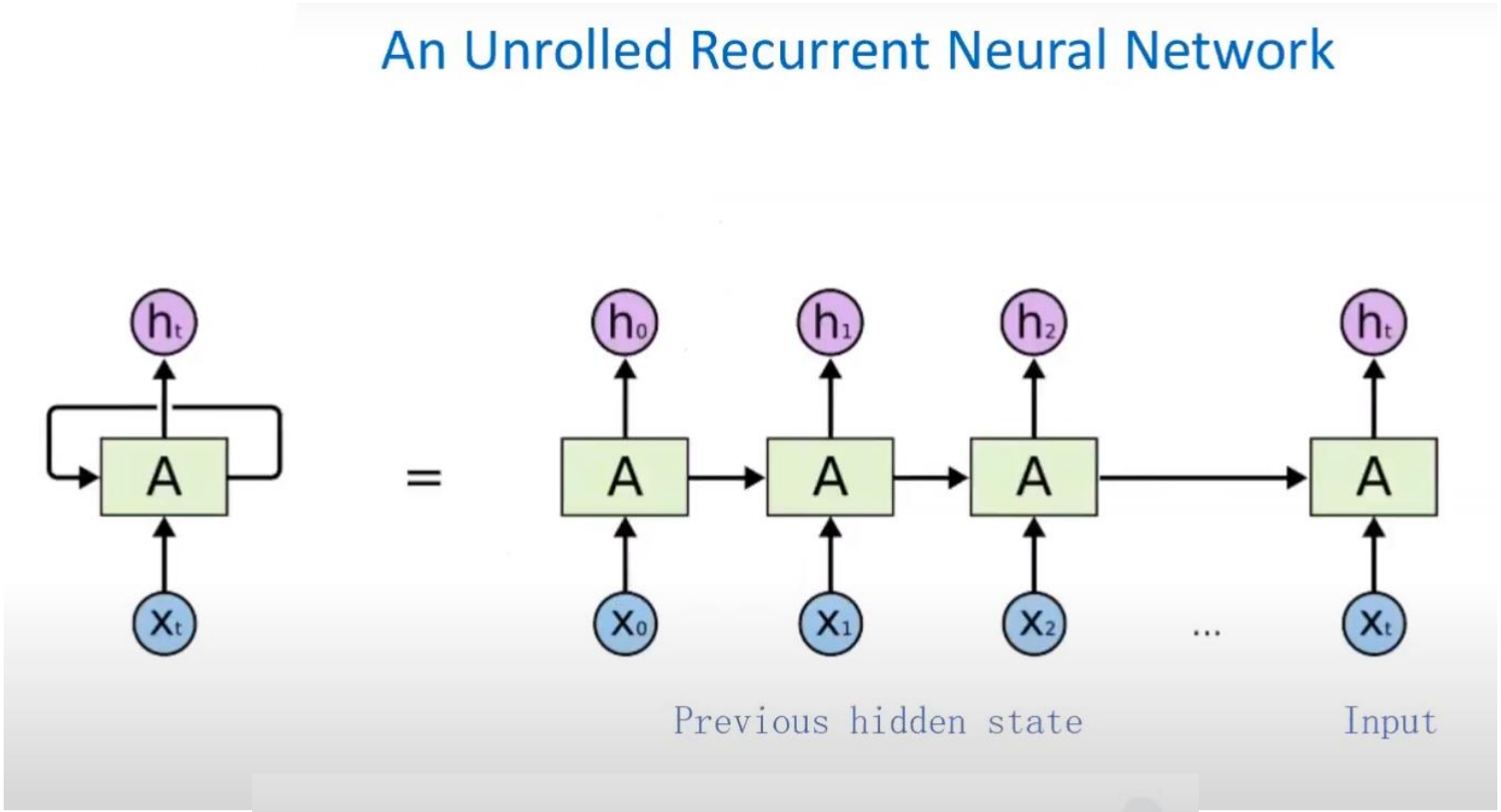
Klassifizieren von Emails mit Method-LSTM(Long Short-Term Memory)

Vorträger : Weifan Zhang
Datum : 19.08.2020

Was ist NN(neural networks)?



Was ist RNN(Recurrent Neural Networks)?

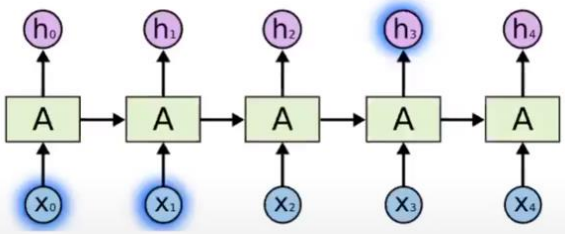


$$h_t = \tanh(W_x * x_t + W_h * h_{t-1} + b)$$

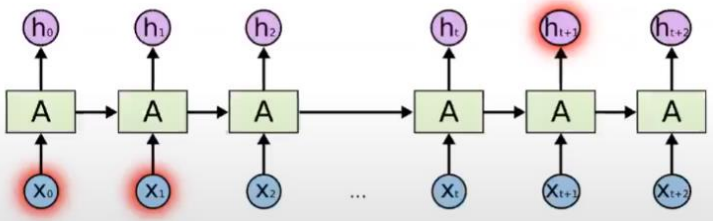
Nachteile von RNN

Cannot Capture Long-term Dependencies

If we want to **predict the last word** in the sentence *"The grass is **green**"*, that's totally doable.



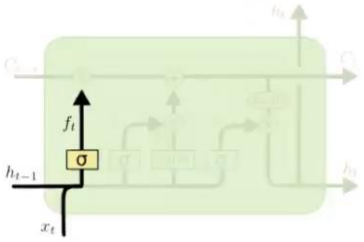
But if we want to **predict the last word** in the sentence *"I am French (2000 words later) I speak fluent **French**"*.
We need to be able to remember long range dependencies.
RNN's are bad at this. They forget the long term past easily.



Eine Lösung-LSTM(Long Short-Term Memory)

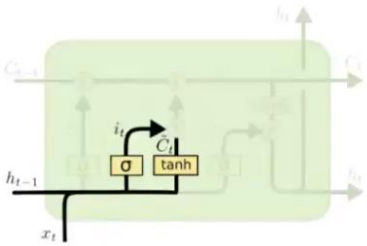
Step-by-Step LSTM Walk Through

1. Forget gate layer



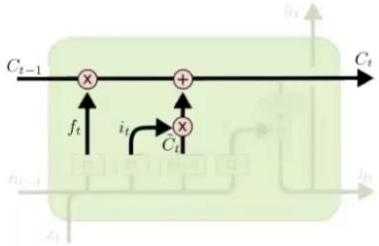
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

2. Input gate layer



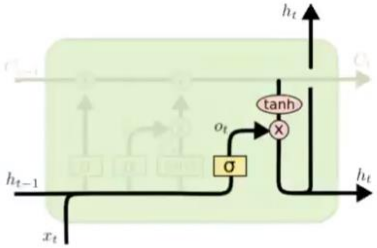
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

3. The current state



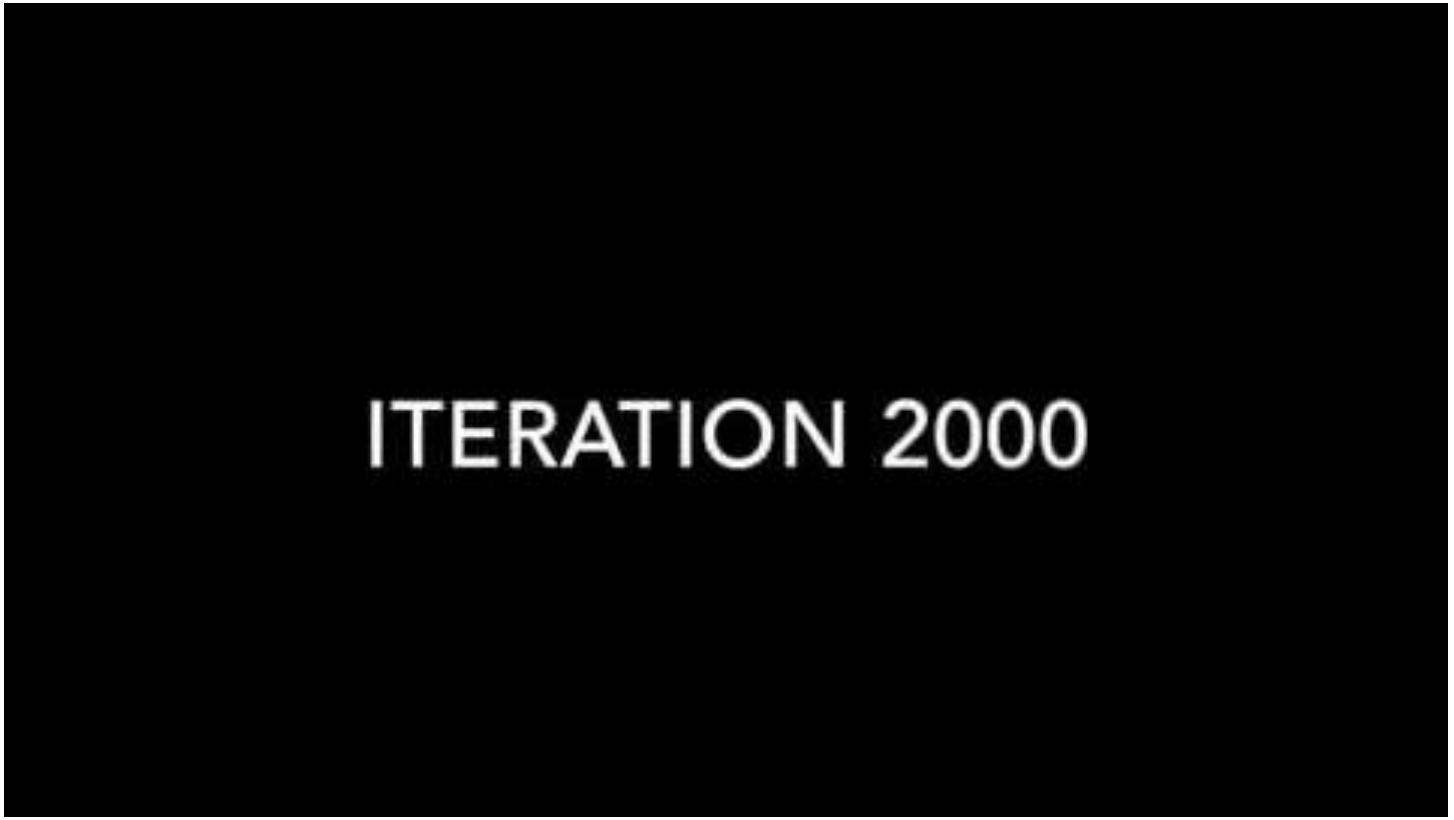
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

4. Output layer



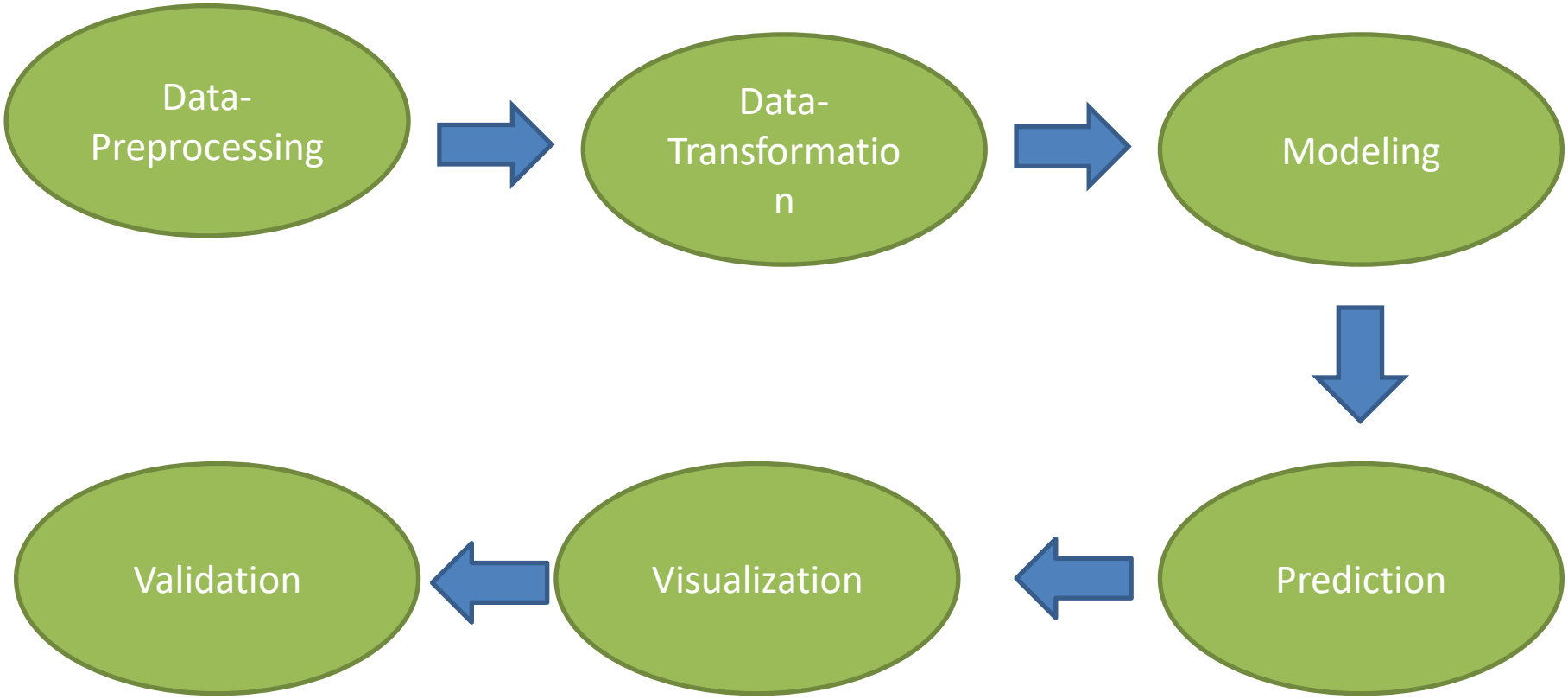
$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Anwendungsbeispiel-Music Generator



Klassifizieren von Emails mit LSTM

Verfahren



Data-Preprocessing

- Alle Inhalte In eine Zeil
- Unwichtiges Zeichen entfernen
- Label einfügen

Subject: vastar resources , inc .
gary , production from the high island larger block a - 1 # 2 commenced on
saturday at 2 : 00 p . m . at about 6 , 500 gross . carlos expects between 9 , 500 and
10 , 000 gross for tomorrow . vastar owns 68 % of the gross production .
george x 3 - 6992
- forwarded by george weissman / hou / ect on 12 / 13 / 99 10 : 16
am -
daren j farmer
12 / 10 / 99 10 : 38 am
to : carlos j rodriguez / hou / ect @ ect
cc : george weissman / hou / ect @ ect , melissa graves / hou / ect @ ect
subject : vastar resources , inc .
carlos ,
please call linda and get everything set up .
i ' m going to estimate 4 , 500 coming up tomorrow , with a 2 , 000 increase each
following day based on my conversations with bill fischer at bmar .
d .
- forwarded by daren j farmer / hou / ect on 12 / 10 / 99 10 : 34
am -
enron north america corp .
from : george weissman 12 / 10 / 99 10 : 00 am
to : daren j farmer / hou / ect @ ect
cc : gary bryan / hou / ect @ ect , melissa graves / hou / ect @ ect
subject : vastar resources , inc .
darren ,
the attached appears to be a nomination from vastar resources , inc . for the
high island larger block a - 1 # 2 (previously , erroneously referred to as the
1 well) . vastar now expects the well to commence production sometime
tomorrow . i told linda harris that we ' d get her a telephone number in gas
control so she can provide notification of the turn - on tomorrow . linda ' s
numbers , for the record , are 281 . 584 . 3359 voice and 713 . 312 . 1689 fax .
would you please see that someone contacts linda and advises her how to
submit future nominations via e - mail , fax or voice ? thanks .
george x 3 - 6992
- forwarded by george weissman / hou / ect on 12 / 10 / 99 09 : 44
am -

	email	label
3094	Subject: calpine 1465ricky sent the nom over e...	1
3296	Subject: guadalupe power partnerstexas indepen...	1
1012	Subject: nom change on tennessee forwarded by ...	1
4948	Subject: dating service for nauuughty minded p...	0
1147	Subject: hpl noms for july 8 2000(see attache...	1

Data-Transformation

```
from keras.preprocessing.text import Tokenizer
tokenizer = Tokenizer(num_words=1000)
tokenizer.fit_on_texts(data['email'].values)
sequences = tokenizer.texts_to_sequences(data['email'].values)
sequences
```

Text zu Sequenz umwanden

```
 105,
 30,
 391,
 59,
 382],
[14, 17, 313, 5, 249, 118, 118, 412, 288, 121, 17, 35, 221],
[14,
 227,
 396,
 139,
 457,
 4,
```

Zweidimensionales Array mit maximaler Sequenz = 20

```
maxlen = 20
from keras import preprocessing
x_train = preprocessing.sequence.pad_sequences(x_train, maxlen=maxlen)
x_test = preprocessing.sequence.pad_sequences(x_test, maxlen=maxlen)
x_train
```

```
array([[331,  44, 796, ..., 623,   7,  44],
       [726,  38, 568, ..., 156, 226,  31],
       [ 46,  17, 672, ..., 126,  46,  17],
       ...,
       [317, 441,  55, ...,  52, 102, 154],
       [ 62, 189,  56, ..., 834,   7,  31],
       [160,  94,   4, ...,   3,  17,  31]], dtype=int32)
```

Modeling

```
from keras.models import Sequential
from keras.layers import Flatten, Dense
from keras.layers import Embedding,LSTM
model = Sequential()
model.add(Embedding(2000, 8, input_length=maxlen))
model.add(LSTM(100, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
model.summary()
history = model.fit(x_train, y_train,
                    epochs=5,
                    batch_size=32,
                    validation_split=0.2)
```

Um Overfitting zu vermeiden

Besser für binäre Daten

Die Best Optimizer nach der Erfahrung

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_2 (Embedding)      (None, 20, 8)             16000
_________________________________________________________________
lstm_2 (LSTM)                (None, 100)               43600
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 101
=================================================================
Total params: 59,701
Trainable params: 59,701
Non-trainable params: 0
_________________________________________________________________
Epoch 1/5
104/104 [==============================] - 4s 41ms/step - loss: 0.4740 - accuracy: 0.7809 - val_loss: 0.2983 - val_accuracy: 0.9118
Epoch 2/5
104/104 [==============================] - 4s 37ms/step - loss: 0.2095 - accuracy: 0.9217 - val_loss: 0.1588 - val_accuracy: 0.9324
Epoch 3/5
104/104 [==============================] - 4s 38ms/step - loss: 0.1538 - accuracy: 0.9426 - val_loss: 0.1470 - val_accuracy: 0.9432
Epoch 4/5
104/104 [==============================] - 4s 37ms/step - loss: 0.1140 - accuracy: 0.9604 - val_loss: 0.1380 - val_accuracy: 0.9444
Epoch 5/5
104/104 [==============================] - 4s 37ms/step - loss: 0.0913 - accuracy: 0.9692 - val_loss: 0.1654 - val_accuracy: 0.9372
```

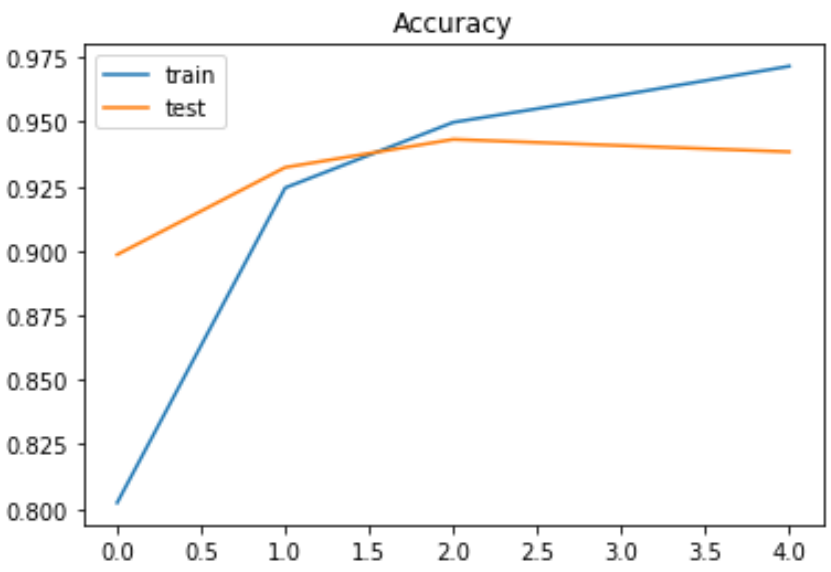
Prediction

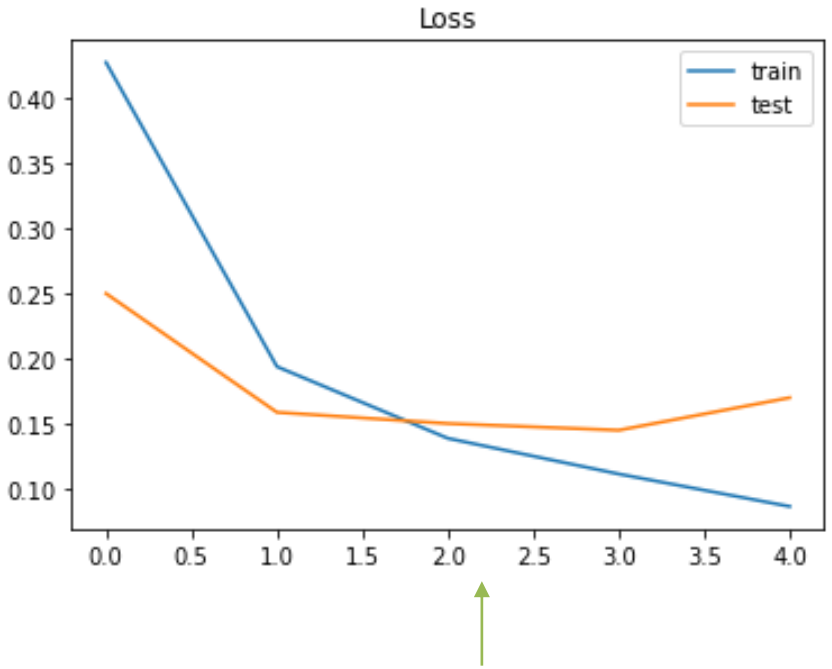
80% als Train - 20% als Test

```
result=model.evaluate(x_test,y_test)
print("test loss: {}\ntest accuracy: {}".format(result[0],result[1]))
```

```
33/33 [==============================] - 0s 5ms/step - loss: 0.1626 - accuracy: 0.9333
test loss:0.16258470714092255
test accuracy:0.9333333373069763
```

Visualization + Validation





Kein Overfitting möglich

Reference

https://en.wikipedia.org/wiki/Artificial_neural_network
https://www.youtube.com/watch?v=EC3SvfW0Z_A
https://colah.github.io/posts/2015-08-Understanding-LSTMs/
https://youtu.be/A2gyidoFsoI
https://blog.csdn.net/weixin_39703655/article/details/104101084

Vielen Dank!