# Sequence-to-Sequence mit neuronalen Netzen

Fynn Schröder

21.06.2018

# Motivation

# Aufgabenstellung

Umwandeln einer Sequenz in eine andere Sequenz (beliebige Längen!)

- Spracherkennung
- Übersetzung
- *Question answering*

# Aufgabenstellung

Umwandeln einer Sequenz in eine andere Sequenz (beliebige Längen!)

- Spracherkennung
- Übersetzung
- *Question answering*

Beispiel 1:

- Eingabe: "*Dies ist ein kurzer Beispielsatz.*" — 5 Wörter
- Ausgabe: "*This is a short example sentence.*" — 6 Wörter

# Aufgabenstellung

Umwandeln einer Sequenz in eine andere Sequenz (beliebige Längen!)

- Spracherkennung
- Übersetzung
- *Question answering*

Beispiel 1:

- Eingabe: "*Dies ist ein kurzer Beispielsatz.*" — 5 Wörter
- Ausgabe: "*This is a short example sentence.*" — 6 Wörter

Beispiel 2:

- Eingabe: `laufen` & *1. Person Präteritum* — 6 Zeichen
- Ausgabe: `lief` — 4 Zeichen

# Erinnerung: RNN

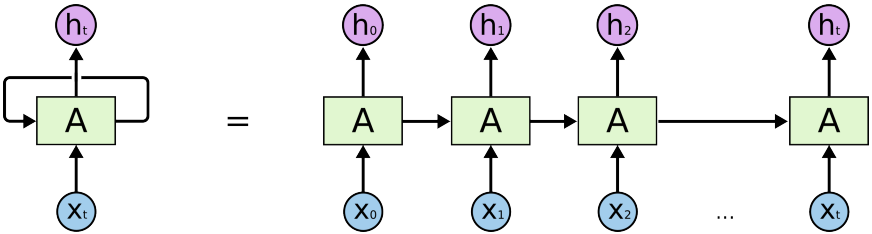


Abbildung 1: Abwicklung eines RNNs in Zeitschritten [1]

# Mögliche Ein-/Ausgabe Szenarien

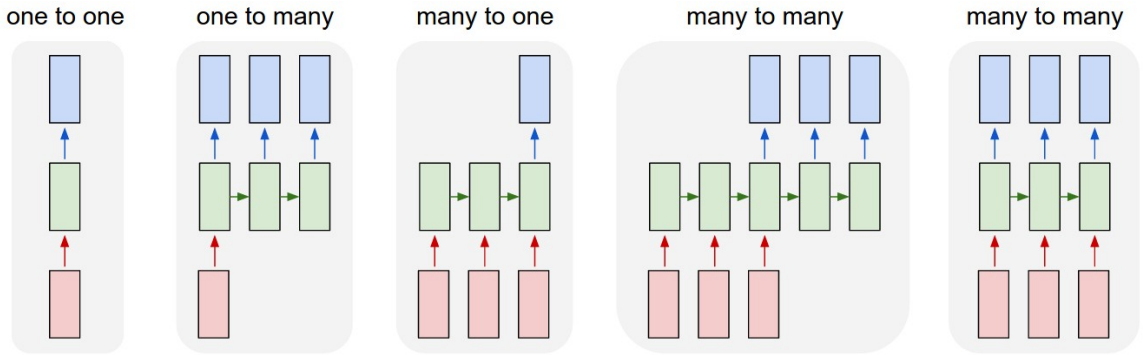


Abbildung 2: RNN Paradigmen [2]

# Deep Neural Networks

Vorteile:

- *State-of-the-art* für viele Probleme im Bereich maschinelles Lernen
  - Bildverarbeitung: Objekterkennung, Segmentierung etc.
  - Diverse weitere Klassifizierungsaufgaben

# Deep Neural Networks

Vorteile:

- *State-of-the-art* für viele Probleme im Bereich maschinelles Lernen
  - Bildverarbeitung: Objekterkennung, Segmentierung etc.
  - Diverse weitere Klassifizierungsaufgaben
- Flexibel und mächtig
  - Features lernen statt händisch entwickeln
  - Einfaches Training per *supervised backpropagation*

# Deep Neural Networks

Vorteile:

- *State-of-the-art* für viele Probleme im Bereich maschinelles Lernen
  - Bildverarbeitung: Objekterkennung, Segmentierung etc.
  - Diverse weitere Klassifizierungsaufgaben
- Flexibel und mächtig
  - Features lernen statt händisch entwickeln
  - Einfaches Training per *supervised backpropagation*

Problem:

- Erfordert Ein- & Ausgabe als Vektoren mit festen Dimensionen [3]

# Sequence to Sequence [3] / Encoder-Decoder Modell [4]

# Architektur

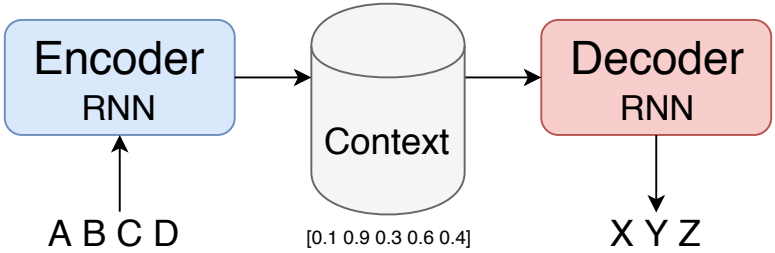


Abbildung 3: Basis Encoder-Decoder Architektur

# Prozess

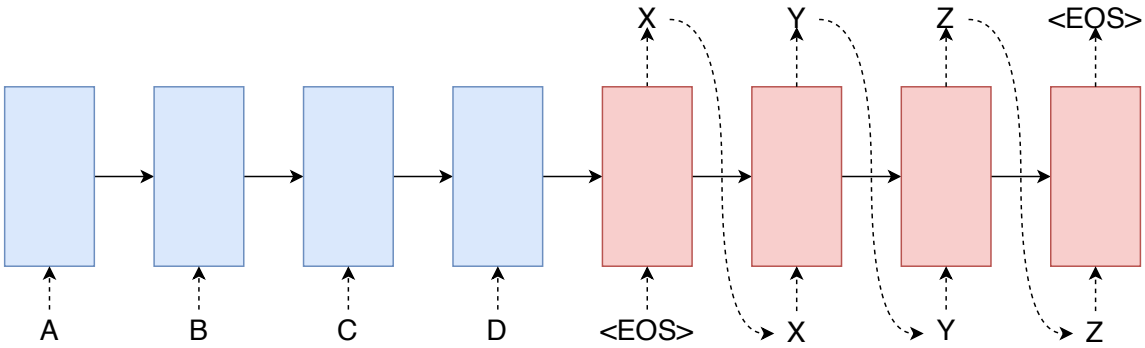


Abbildung 4: Encoder-Decoder Verarbeitung

# Pseudo-Quellcode

```
encoder = EncoderRNN(input_size=26, embed_size=16, hidden_size=32)
decoder = DecoderRNN(output_size=27, embed_size=16, hidden_size=32)

encoder_outputs, encoder_hidden = encoder(input_tensor)
decoder_hidden = encoder_hidden
decoder_input = END_OF_SEQUENZ.clone()

for i in range(target_length):
    decoder_output, decoder_hidden = decoder(decoder_input, decoder_hidden)
    decoder_input = most_likely_element(decoder_output)
```

# Eigenschaften

Vorteile

- Entkopplung von Ein- und Ausgabe
- Encoder reduziert Tensor beliebiger Dimension auf Tensor fester Größe
- Decoder kann beliebig oft nach weiterem Element der Ausgabesequenz gefragt werden

# Eigenschaften

Vorteile

- Entkopplung von Ein- und Ausgabe
- Encoder reduziert Tensor beliebiger Dimension auf Tensor fester Größe
- Decoder kann beliebig oft nach weiterem Element der Ausgabesequenz gefragt werden

Nachteile

- Hidden-State des Encoders $\rightarrow$ initialer Hidden-State des Decoder
  - Gesamte Eingabesequenz im Context-Tensor
  - Lange Sequenzen problematisch: Langzeitabhängigkeiten
  - Decoder hat nur Zugriff auf "Gesamtbedeutung" der Eingabesequenz, kein Fokus auf einzelne Elemente

# Langzeitabhängigkeiten

- Eingabe: *"He has **gone** home after work."*
- Ziel Ausgabe: *"Er ist nach der Arbeit nach Hause **gegangen**."*

# Langzeitabhängigkeiten

- Eingabe: *"He has **gone** home after work."*
- Ziel Ausgabe: *"Er ist nach der Arbeit nach Hause **gegangen**."*

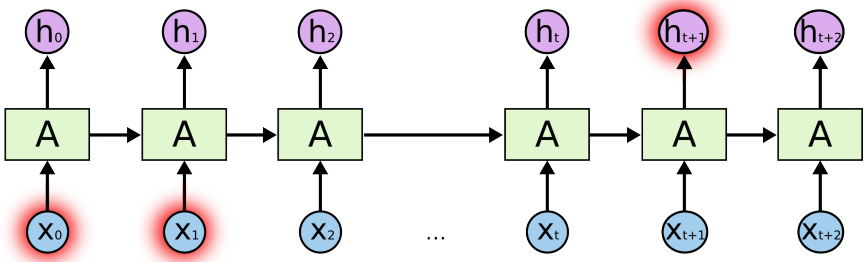


Abbildung 5: RNN mit Langzeitabhängigkeiten [1]

# Verbesserungen

# LSTM / GRU statt RNN

- LSTM (*Long sort-term memory*): Kann erfolgreich lange temporale Abhängigkeiten lernen
- GRU (*Gated recurrent unit*): Ähnliche Fähigkeiten wie LSTM – einfacher, kein *Output-Gate*

# LSTM / GRU statt RNN

- LSTM (*Long sort-term memory*): Kann erfolgreich lange temporale Abhängigkeiten lernen
- GRU (*Gated recurrent unit*): Ähnliche Fähigkeiten wie LSTM – einfacher, kein *Output-Gate*

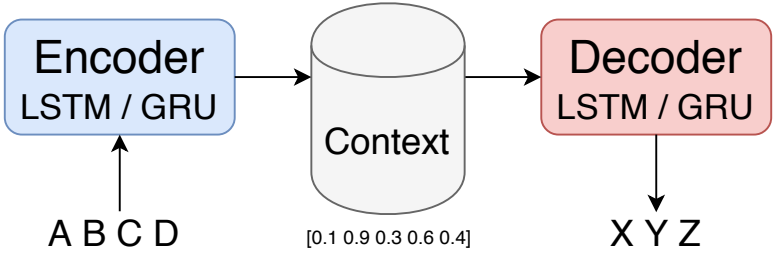


Abbildung 6: Encoder-Decoder Architektur mit LSTM / GRU

# Zugriff auf Eingabe-Kontext

- Ein- und Ausgabe vorher alignieren
- Decoder erhält für jede Ausgabe genau eine Eingabe als Kontext

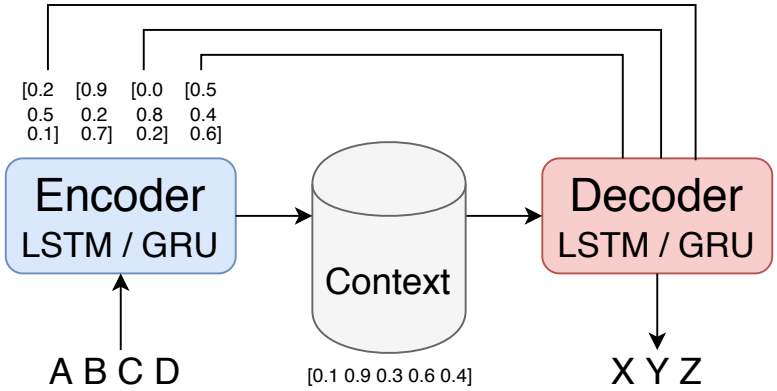


Abbildung 7: Encoder-Decoder Architektur mit festem Eingabe-Kontext

# Attention

- Alignment-Modell [5]: Wie gut passen welche Eingaben zur aktuellen Ausgabe?

# Attention

- Alignment-Modell [5]: Wie gut passen welche Eingaben zur aktuellen Ausgabe?
- Steuert den Zugriff des Decoders auf alle Ausgaben des Encoders

# Attention

- Alignment-Modell [5]: Wie gut passen welche Eingaben zur aktuellen Ausgabe?
- Steuert den Zugriff des Decoders auf alle Ausgaben des Encoders
- Training per *Backpropagation* als ganzes Netzwerk

# Attention

- Alignment-Modell [5]: Wie gut passen welche Eingaben zur aktuellen Ausgabe?
- Steuert den Zugriff des Decoders auf alle Ausgaben des Encoders
- Training per *Backpropagation* als ganzes Netzwerk
- Fokussierung auf notwendigen Informationen der aktuellen Ausgabe

# Attention

- Alignment-Modell [5]: Wie gut passen welche Eingaben zur aktuellen Ausgabe?
- Steuert den Zugriff des Decoders auf alle Ausgaben des Encoders
- Training per *Backpropagation* als ganzes Netzwerk
- Fokussierung auf notwendigen Informationen der aktuellen Ausgabe
- Zuverlässig auch für lange Sequenzen

# Attention

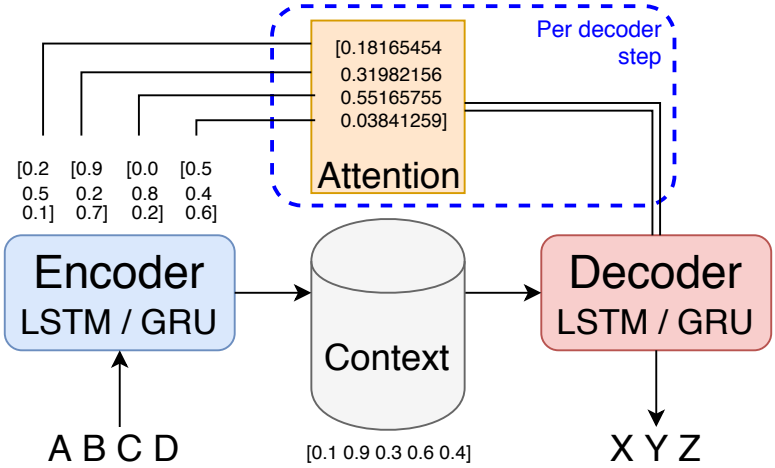


Abbildung 8: Encoder-Decoder Architektur mit Attention

# Pseudo-Quellcode

```
encoder = EncoderRNN(input_size=26, embed_size=16, hidden_size=32)
decoder = AttentionDecoderRNN(output_size=27, embed_size=16, hidden_size=32)

encoder_outputs, encoder_hidden = encoder(input_tensor)
decoder_hidden = encoder_hidden
decoder_input = END_OF_SEQUENZ.clone()

for i in range(target_length):
    decoder_output, decoder_hidden = decoder(decoder_input,
                                             decoder_hidden,
                                             encoder_outputs)
    decoder_input = most_likely_element(decoder_output)
```

# Ergebnisse

Tabelle 1: Entwicklung des BLEU Scores im WMT NewsTest 2014

| Jahr | Modell | EN-FR | EN-DE |
|---|---|---|---|
| 2014 | Baseline: Phrasenbasiert + LM | 33.30 | |
| 2014 | Sieger WMT14: Phrasenbasiert + LM [6] | | 20.70 |
| 2014 | Encoder-Decoder RNN (1) [4] | 17.82 | |
| 2014 | Encoder-Decoder RNN (1) + Attention [4] | 26.75 | |
| 2015 | Encoder-Decoder RNN (2) [7] | | 11.30 |
| 2015 | Encoder-Decoder RNN (2) + Attention [7] | | 20.80 |
| 2016 | ByteNet (Enc-Dec, CNN, Attention) [8] | | 23.75 |
| 2016 | Google's Neural Machine Translation [9] | 38.95 | 24.67 |

# Ausblick

# Attention Is All You Need [10]

- Neues Modell “Transformer”

# Attention Is All You Need [10]

- Neues Modell “Transformer”
- *Multi-headed self-attention*: Beziehungen zwischen verschiedenen Positionen zur Berechnung der Representation der Sequenz

# Attention Is All You Need [10]

- Neues Modell “Transformer”
- *Multi-headed self-attention*: Beziehungen zwischen verschiedenen Positionen zur Berechnung der Representation der Sequenz
- Verwendet “*positional encodings*” statt Rekurrenz oder *Convolution*

# Attention Is All You Need [10]

- Neues Modell “Transformer”
- *Multi-headed self-attention*: Beziehungen zwischen verschiedenen Positionen zur Berechnung der Representation der Sequenz
- Verwendet “*positional encodings*” statt Rekurrenz oder *Convolution*
- Merklich schnelleres Training als Encoder-Decoder RNNs

# Attention Is All You Need [10]

- Neues Modell “Transformer”
- *Multi-headed self-attention*: Beziehungen zwischen verschiedenen Positionen zur Berechnung der Representation der Sequenz
- Verwendet “*positional encodings*” statt Rekurrenz oder *Convolution*
- Merklich schnelleres Training als Encoder-Decoder RNNs
- Neue Bestleistung in WMT 2014 English-to-German und English-to-French

# Ergebnisse

Tabelle 2: Entwicklung des BLEU Scores im WMT NewsTest 2014

| Jahr | Modell | EN-FR | EN-DE |
|---|---|---|---|
| 2014 | Baseline: Phrasenbasiert + LM | 33.30 | |
| 2014 | Sieger WMT14: Phrasenbasiert + LM [6] | | 20.70 |
| 2014 | Encoder-Decoder RNN (1) [4] | 17.82 | |
| 2014 | Encoder-Decoder RNN (1) + Attention [4] | 26.75 | |
| 2015 | Encoder-Decoder RNN (2) [7] | | 11.30 |
| 2015 | Encoder-Decoder RNN (2) + Attention [7] | | 20.80 |
| 2016 | ByteNet (Enc-Dec, CNN, Attention) [8] | | 23.75 |
| 2016 | Google's Neural Machine Translation [9] | 38.95 | 24.67 |
| 2017 | Transformer [10] | 41.80 | 28.40 |

# Quellen

# References I

[1] C. Olah, „Understanding LSTM Networks“. [Online]. Verfügbar unter: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[2] A. Karpathy, „The Unreasonable Effectiveness of Recurrent Neural Networks“. [Online]. Verfügbar unter: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

[3] I. Sutskever, O. Vinyals, und Q. V. Le, „Sequence to sequence learning with neural networks“, in *Advances in neural information processing systems*, 2014, S. 3104–3112.

[4] K. Cho, B. van Merrienboer, Çaglar Gülçehre, F. Bougares, H. Schwenk, und Y. Bengio, „Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation“, *CoRR*, Bd. abs/1406.1078, 2014 [Online]. Verfügbar unter: http://arxiv.org/abs/1406.1078

# References II

[5] D. Bahdanau, K. Cho, und Y. Bengio, „Neural Machine Translation by Jointly Learning to Align and Translate“, *CoRR*, Bd. abs/1409.0473, 2014 [Online]. Verfügbar unter: http://arxiv.org/abs/1409.0473

[6] M. Freitag *u. a.*, „EU-BRIDGE MT: Combined Machine Translation“, in *WMT@ACL*, 2014.

[7] M. Luong, H. Pham, und C. D. Manning, „Effective Approaches to Attention-based Neural Machine Translation“, *CoRR*, Bd. abs/1508.04025, 2015 [Online]. Verfügbar unter: http://arxiv.org/abs/1508.04025

[8] N. Kalchbrenner, L. Espeholt, K. Simonyan, A. van den Oord, A. Graves, und K. Kavukcuoglu, „Neural Machine Translation in Linear Time“, *CoRR*, Bd. abs/1610.10099, 2016 [Online]. Verfügbar unter: http://arxiv.org/abs/1610.10099

# References III

[9] Y. Wu *u. a.*, „Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation“, *CoRR*, Bd. abs/1609.08144, 2016 [Online]. Verfügbar unter: http://arxiv.org/abs/1609.08144

[10] A. Vaswani *u. a.*, „Attention Is All You Need“, *CoRR*, Bd. abs/1706.03762, 2017 [Online]. Verfügbar unter: http://arxiv.org/abs/1706.03762